Arria 10 Hard Processor System Technical Reference Manual

Last updated for Quartus Prime Design Suite: 16.1
a10_5v4
2016.10.28

101 Innovation Drive
San Jose, CA 95134
www.altera.com

Subscribe
Send Feedback

now part of Intel

Contents

Introduction to the Hard Processor System..... 1-1
Features of the HPS..... 1-3
HPS Block Diagram and System Integration..... 1-5
HPS Block Diagram..... 1-5
Cortex-A9 MPCore..... 1-6
HPS Interfaces..... 1-6
System Interconnect..... 1-7
On-Chip Memory..... 1-8
Flash Memory Controllers..... 1-9
Support Peripherals..... 1-10
Interface Peripherals..... 1-13
CoreSight Debug and Trace..... 1-16
Hard Processor System I/O Pin Multiplexing..... 1-16
Endian Support..... 1-16
Introduction to the Hard Processor System Address Map..... 1-17
HPS Address Spaces..... 1-17
HPS Peripheral Region Address Map..... 1-20
Document Revision History..... 1-23

Clock Manager..... 2-1
Features of the Clock Manager..... 2-1
Clock Manager Block Diagram and System Integration..... 2-3
L4 Peripheral Clocks..... 2-5
Boot Clock..... 2-6
Functional Description of the Clock Manager..... 2-7
Clock Manager Building Blocks..... 2-7
PLL Integration..... 2-9
Hardware-Managed and Software-Managed Clocks..... 2-11
Hardware Sequenced Clock Groups..... 2-11
Software Sequenced Clocks..... 2-14
Resets..... 2-16
Security..... 2-17
Interrupts..... 2-18
Clock Manager Address Map and Register Definitions..... 2-18
clkmgr_clkmgrp Address Map..... 2-18
clkmgr_mainpllgrp Address Map..... 2-39
clkmgr_perpllgrp Address Map..... 2-67
clkmgr_alteragrp Address Map..... 2-96
Document Revision History..... 2-99

Reset Manager..... 3-1

Reset Manager Block Diagram and System Integration..... 3-3
Reset Controller.....3-4
Security Reset Functions..... 3-8
Module Reset Signals..... 3-8
Slave Interface and Status Register..... 3-14
Functional Description of the Reset Manager..... 3-15
Reset Sequencing.....3-16
Reset Pins..... 3-20
Reset Effects..... 3-21
Reset Handshaking..... 3-21
Reset Manager Address Map and Register Definitions.....3-21
rst_mgr Address Map..... 3-21
Document Revision History.....3-73

FPGA Manager..... 4-1
Features of the FPGA Manager..... 4-1
FPGA Manager Block Diagram and System Integration.....4-2
Functional Description of the FPGA Manager..... 4-3
FPGA Configuration..... 4-3
FPGA Status.....4-4
Error Message Extraction.....4-4
Data AES Decryption..... 4-4
Boot Handshake..... 4-4
General Purpose I/O.....4-5
Clock..... 4-5
Reset..... 4-5
FPGA Manager Address Map and Register Definitions..... 4-5
fpga_mgr_fpgamgrdata Address Map.....4-6
fpga_mgr_fpgamgrregs Address Map..... 4-7
Document Revision History.....4-49

System Manager.....5-1
Features of the System Manager.....5-1
System Manager Block Diagram and System Integration..... 5-2
Functional Description of the System Manager.....5-3
Boot Configuration and System Information..... 5-3
Additional Module Control..... 5-3
Boot ROM Code.....5-6
FPGA Interface Enables..... 5-7
ECC and Parity Control..... 5-8
Preloader Handoff Information..... 5-8
Clocks..... 5-8
Resets..... 5-8
System Manager Address Map and Register Definitions.....5-9
sys_mgr_core Address Map.....5-9
sys_mgr_rom Address Map.....5-90
Document Revision History.....5-107

SoC Security....6-1
Security Manager....6-2
Security Manager Block Diagram....6-3
Functional Overview....6-4
Functional Description....6-4
Security System Extensions....6-21
TrustZone....6-21
Arria 10 HPS Secure Firewalls....6-24
Revision History....6-37

System Interconnect....7-1
About the System Interconnect....7-2
Features of the System Interconnect....7-2
System Interconnect Block Diagram and System Integration....7-2
Arria 10 HPS Secure Firewalls....7-7
About the Rate Adapters....7-8
About the SDRAM L3 Interconnect....7-8
About Quality of Service and Arbitration....7-11
About the Service Network....7-11
About the Observation Network....7-11
Functional Description of the System Interconnect....7-12
System Interconnect Address Spaces....7-13
Secure Transaction Protection....7-20
System Interconnect Master Properties....7-20
System Interconnect Slave Properties....7-22
System Interconnect Clocks....7-24
System Interconnect Resets....7-26
Functional Description of the Rate Adapters....7-26
Functional Description of the Firewalls....7-27
Functional Description of the SDRAM L3 Interconnect....7-28
Functional Description of the QoS Generators....7-34
Functional Description of the Arbitration Logic....7-39
Functional Description of the Observation Network....7-40
Configuring the System Interconnect....7-42
Configuring the Rate Adapters....7-42
Configuring the SDRAM Scheduler....7-42
Configuring the Hard Memory Controller....7-43
System Interconnect Address Map and Register Definitions....7-45
io48_hmc_mmr Address Map....7-45
ecc_hmc_ocp_slv_block Address Map....7-126
noc_l4_priv_l4_priv_filter Address Map....7-189
noc_mpu_m0_MPU_M1toDDRResp_main_RateAdapter Address Map....7-203
noc_mpu_m0_MPU_M0_rate_adResp_main_RateAdapter Address Map....7-208
noc_mpu_m0_L4_MP_rate_ad_main_RateAdapter Address Map....7-213
noc_mpu_m0_fpga2soc_rate_ad_main_RateAdapter Address Map....7-218
noc_mpu_m0_L3Tosoc2fpgaResp_main_RateAdapter Address Map....7-223

noc_mpu_m0_acp_rate_ad_main_RateAdapter Address Map 7-228
noc_mpu_m0_ddr_T_main_Probe Address Map 7-233
noc_mpu_m0_ddr_T_main_Scheduler Address Map 7-302
noc_fw_l4_per_l4_per_scr Address Map 7-312
noc_fw_l4_sys_l4_sys_scr Address Map 7-357
noc_fw_ocram_ocram_scr Address Map 7-411
noc_fw_ddr_mpu_fpga2sdram_ddr_scr Address Map 7-423
noc_fw_ddr_l3_ddr_scr Address Map 7-449
noc_fw_soc2fpga_soc2fpga_scr Address Map 7-464
noc_mpu_m0_Probe_SoC2FPGA_main_Probe Address Map 7-470
noc_mpu_m0_Probe_emacs_main_Probe Address Map 7-530
noc_mpu_m0_Probe_emacs_main_TransactionStatProfiler Address Map 7-590
noc_mpu_m0_cs_obs_at_main_AtbEndPoint Address Map 7-603
noc_mpu_m0_cs_obs_at_main_ErrorLogger_0 Address Map 7-608
noc_mpu_m0_Probe_MPU_main_Probe Address Map 7-619
noc_mpu_m0_I_main_QosGenerator Address Map 7-663
noc_mpu_m1_I_main_QosGenerator Address Map 7-672
noc_mpu_fpga2soc_axi32_I_main_QosGenerator Address Map 7-681
noc_mpu_fpga2soc_axi64_I_main_QosGenerator Address Map 7-690
noc_mpu_fpga2soc_axi128_I_main_QosGenerator Address Map 7-699
noc_mpu_dma_I_main_QosGenerator Address Map 7-708
noc_mpu_emac0_m_I_main_QosGenerator Address Map 7-717
noc_mpu_emac1_m_I_main_QosGenerator Address Map 7-726
noc_mpu_emac2_m_I_main_QosGenerator Address Map 7-735
noc_mpu_usb0_m_I_main_QosGenerator Address Map 7-744
noc_mpu_usb1_m_I_main_QosGenerator Address Map 7-753
noc_mpu_nand_m_I_main_QosGenerator Address Map 7-762
noc_mpu_sdmmc_m_I_main_QosGenerator Address Map 7-771
noc_mpu_fpga2sdram0_axi32_I_main_QosGenerator Address Map 7-780
noc_mpu_fpga2sdram0_axi64_I_main_QosGenerator Address Map 7-789
noc_mpu_fpga2sdram0_axi128_I_main_QosGenerator Address Map 7-798
noc_mpu_fpga2sdram1_axi32_I_main_QosGenerator Address Map 7-807
noc_mpu_fpga2sdram1_axi64_I_main_QosGenerator Address Map 7-816
noc_mpu_fpga2sdram2_axi32_I_main_QosGenerator Address Map 7-825
noc_mpu_fpga2sdram2_axi64_I_main_QosGenerator Address Map 7-834
noc_mpu_fpga2sdram2_axi128_I_main_QosGenerator Address Map 7-843
noc_mpu_emac0_m_I_main_TransactionStatFilter Address Map 7-852
Document Revision History 7-864

HPS-FPGA Bridges 8-1
Features of the HPS-FPGA Bridges 8-1
Arria 10 HPS-FPGA Bridges Block Diagram and System Integration 8-3
Functional Description of the HPS-FPGA Bridges 8-3
Functional Description of the FPGA-to-HPS Bridge 8-3
Functional Description of the HPS-to-FPGA Bridge 8-7
Functional Description of the Lightweight HPS-to-FPGA Bridge 8-10
Clocks and Resets 8-14
Data Width Sizing 8-15

Ready Latency Support....8-15
HPS-FPGA Bridges Address Map and Register Definitions for Arria 10....8-15
fpga_bridge_soc2fpga128 Address Map....8-15
fpga_bridge_lwsoc2fpga Address Map....8-16
Document Revision History....8-16

Cortex-A9 Microprocessor Unit Subsystem....9-1
Features of the Cortex-A9 MPU Subsystem....9-1
Cortex-A9 MPU Subsystem Block Diagram and System Integration....9-2
Cortex-A9 MPU Subsystem with System Interconnect....9-2
Cortex-A9 MPU Subsystem Internals....9-3
Cortex-A9 MPCore....9-4
Functional Description....9-4
Implementation Details....9-5
Cortex-A9 Processor....9-6
Interactive Debugging Features....9-7
L1 Caches....9-7
Preload Engine....9-7
Floating Point Unit....9-8
NEON Multimedia Processing Engine....9-9
Memory Management Unit....9-10
Performance Monitoring Unit....9-12
ARM Cortex-A9 MPCore Timers....9-12
Generic Interrupt Controller....9-13
Global Timer....9-27
Snoop Control Unit....9-28
Accelerator Coherency Port....9-29
L2 Cache....9-34
Functional Description....9-34
CPU Prefetch....9-41
TrustZone....9-41
Secure Partitioning....9-42
Virtual Processor Operation....9-43
Secure Debug....9-44
Debugging Modules....9-44
Program Trace....9-44
Event Trace....9-45
Cross-Triggering....9-46
Clocks....9-46
Cortex-A9 MPU Subsystem Register Implementation....9-46
Cortex-A9 MPU Subsystem Address Map for Arria 10....9-47
L2 Cache Controller Address Map for Arria 10....9-48
Document Revision History....9-50

CoreSight Debug and Trace....10-1
Features of CoreSight Debug and Trace....10-2
ARM CoreSight Documentation....10-2

CoreSight Debug and Trace Block Diagram and System Integration....10-3
Functional Description of CoreSight Debug and Trace....10-4
Debug Access Port....10-4
System Trace Macrocell....10-5
Trace Funnel....10-5
CoreSight Trace Memory Controller....10-6
AMBA Trace Bus Replicator....10-7
Trace Port Interface Unit....10-7
Embedded Cross Trigger System....10-7
Program Trace Macrocell....10-12
HPS Debug APB Interface....10-12
FPGA Interface....10-12
Debug Clocks....10-16
Debug Resets....10-17
CoreSight Debug and Trace Programming Model....10-18
Coresight Component Address....10-18
STM Channels....10-19
CTI Trigger Connections to Outside the Debug System....10-21
Configuring Embedded Cross-Trigger Connections....10-22
CoreSight Debug and Trace Address Map and Register Definitions....10-24
Document Revision History....10-24

Error Checking and Correction Controller....11-1
ECC Controller Features....11-1
ECC Supported Memories....11-1
ECC Controller Block Diagram and System Integration....11-2
ECC Controller Functional Description....11-3
Overview....11-3
ECC Structure....11-3
Memory Data Initialization....11-5
Indirect Memory Access....11-6
Error Logging....11-11
ECC Controller Interrupts....11-13
ECC Controller Initialization and Configuration....11-16
ECC Controller Clocks....11-17
ECC Controller Reset....11-18
ECC Controller Address Map and Register Descriptions....11-18
emac_rx_ecc Address Map....11-18
emac_tx_ecc Address Map....11-63
nandecc_ecc Address Map....11-108
nandr_ecc Address Map....11-140
nandw_ecc Address Map....11-173
sdmmc_ecc Address Map....11-205
onchip_ram_ecc Address Map....11-241
dmac_ecc Address Map....11-274
qspi_ecc Address Map....11-306
usb_ecc Address Map....11-339
Revision History....11-377

On-Chip Memory..... 12-1
On-Chip RAM.....12-1
Features of the On-Chip RAM.....12-1
On-Chip RAM Block Diagram and System Integration.....12-2
Functional Description of the On-Chip RAM.....12-2
Boot ROM.....12-4
Features of the Boot ROM.....12-4
Boot ROM Block Diagram and System Integration.....12-5
Functional Description of the Boot ROM.....12-5
On-Chip Memory Address Map and Register Definitions.....12-6
onchip_ram_ecc Address Map.....12-6
ram_onchip_ram_block Address Map.....12-39
rom_onchip_rom_block Address Map.....12-39
Document Revision History.....12-39

NAND Flash Controller..... 13-1
NAND Flash Controller Features.....13-1
NAND Flash Controller Block Diagram and System Integration.....13-2
NAND Flash Controller Signal Descriptions.....13-2
Functional Description of the NAND Flash Controller.....13-4
Discovery and Initialization.....13-4
Bootstrap Interface.....13-5
Configuration by Host.....13-5
Local Memory Buffer.....13-6
Clocks.....13-7
Resets.....13-7
Indexed Addressing.....13-8
Command Mapping.....13-9
Data DMA.....13-14
ECC.....13-18
NAND Flash Controller Programming Model.....13-21
Basic Flash Programming.....13-22
Flash-Related Special Function Operations.....13-25
NAND Flash Controller Address Map and Register Definitions.....13-33
nandecc_ecc Address Map.....13-33
nandr_ecc Address Map.....13-66
nandw_ecc Address Map.....13-98
nand_config Address Map.....13-131
nand_param Address Map.....13-190
nand_status Address Map.....13-212
nand_ecc Address Map.....13-252
nand_dma Address Map.....13-256
nand_NANDDATA Address Map.....13-276
Document Revision History.....13-276

SD/MMC Controller.....14-1
Features of the SD/MMC Controller.....14-1
SD Card Support Matrix.....14-3
MMC Support Matrix.....14-3
SD/MMC Controller Block Diagram and System Integration.....14-4
SD/MMC Controller Signal Description.....14-4
Functional Description of the SD/MMC Controller.....14-5
SD/MMC/CE-ATA Protocol.....14-5
BIU.....14-6
CIU.....14-20
Clocks.....14-35
Resets.....14-36
Voltage Switching.....14-37
SD/MMC Controller Programming Model.....14-39
Software and Hardware Restrictions†.....14-39
Initialization†.....14-41
Controller/DMA/FIFO Buffer Reset Usage.....14-47
Non-Data Transfer Commands.....14-47
Data Transfer Commands.....14-49
Transfer Stop and Abort Commands.....14-56
Internal DMA Controller Operations.....14-57
Commands for SDIO Card Devices.....14-59
CE-ATA Data Transfer Commands.....14-62
Card Read Threshold.....14-69
Interrupt and Error Handling.....14-72
Booting Operation for eMMC and MMC.....14-74
SD/MMC Controller Address Map and Register Definitions.....14-84
sdmmc Address Map.....14-84
sdmmc_ecc Address Map.....14-180
Document Revision History.....14-216

Quad SPI Flash Controller.....15-1
Features of the Quad SPI Flash Controller.....15-1
Quad SPI Flash Controller Block Diagram and System Integration.....15-2
Quad SPI Flash Controller Signal Description.....15-3
Functional Description of the Quad SPI Flash Controller.....15-4
Overview.....15-4
Data Slave Interface.....15-4
SPI Legacy Mode.....15-8
Register Slave Interface.....15-8
Local Memory Buffer.....15-9
DMA Peripheral Request Controller.....15-10
Arbitration between Direct/Indirect Access Controller and STIG.....15-11
Configuring the Flash Device.....15-11
XIP Mode.....15-13
Write Protection.....15-13

Data Slave Sequential Access Detection 15-13
Clocks 15-13
Resets 15-14
Interrupts 15-14
Quad SPI Flash Controller Programming Model 15-16
Setting Up the Quad SPI Flash Controller 15-16
Indirect Read Operation with DMA Disabled 15-17
Indirect Read Operation with DMA Enabled 15-17
Indirect Write Operation with DMA Disabled 15-18
Indirect Write Operation with DMA Enabled 15-18
XIP Mode Operations 15-19
Quad SPI Flash Controller Address Map and Register Definitions 15-20
qspi_QSPIDATA Address Map 15-21
qspiregs Address Map 15-21
qspi_ecc Address Map 15-77
Document Revision History 15-110

DMA Controller 16-1
Features of the DMA Controller 16-1
DMA Controller Block Diagram and System Integration 16-3
Functional Description of the DMA Controller 16-4
AXI Characteristics for a DMA Transfer 16-5
Operating States 16-6
Initializing the DMAC 16-9
Using the Slave Interfaces 16-10
Peripheral Request Interface 16-11
Using Events and Interrupts 16-22
Aborts 16-24
Security Usage 16-27
Programming Restrictions 16-30
Constraints and Limitations of Use 16-34
DMA Controller Programming Model 16-35
Instruction Syntax Conventions 16-35
Instruction Set Summary 16-35
Instructions 16-37
Assembler Directives 16-52
MFIFO Buffer Usage Overview 16-54
DMA Controller Address Map and Register Definitions 16-62
DMA Controller Address Map and Register Definitions 16-63
Document Revision History 16-99

Ethernet Media Access Controller 17-1
Features of the Ethernet MAC 17-2
MAC 17-2
DMA 17-2
Management Interface 17-3
Acceleration 17-3

PHY Interface 17-3
EMAC Block Diagram and System Integration 17-4
EMAC Signal Description 17-5
HPS EMAC I/O Signals 17-6
FPGA EMAC I/O Signals 17-8
PHY Management Interface 17-11
EMAC Internal Interfaces 17-12
DMA Master Interface 17-12
Timestamp Interface 17-13
System Manager Configuration Interface 17-15
Functional Description of the EMAC 17-16
Transmit and Receive Data FIFO Buffers 17-17
DMA Controller 17-18
Descriptor Overview 17-31
IEEE 1588-2002 Timestamps 17-48
IEEE 1588-2008 Advanced Timestamps 17-54
IEEE 802.3az Energy Efficient Ethernet 17-57
Checksum Offload 17-58
Frame Filtering 17-58
Clocks and Resets 17-63
Interrupts 17-68
Ethernet MAC Programming Model 17-68
System Level EMAC Configuration Registers 17-69
EMAC FPGA Interface Initialization 17-71
EMAC HPS Interface Initialization 17-72
DMA Initialization 17-73
EMAC Initialization and Configuration 17-74
Performing Normal Receive and Transmit Operation 17-75
Stopping and Starting Transmission 17-75
Programming Guidelines for Energy Efficient Ethernet 17-75
Programming Guidelines for Flexible Pulse-Per-Second (PPS) Output 17-77
Ethernet MAC Address Map and Register Definitions 17-78
emac Address Map 17-79
emac_rx_ecc Address Map 17-1224
emac_tx_ecc Address Map 17-1269
Document Revision History 17-1314

USB 2.0 OTG Controller 18-1
Features of the USB OTG Controller 18-2
Supported PHYS 18-3
USB OTG Controller Block Diagram and System Integration 18-4
USB 2.0 ULPI PHY Signal Description 18-5
Functional Description of the USB OTG Controller 18-6
USB OTG Controller Block Description 18-6
Local Memory Buffer 18-10
Clocks 18-10
Resets 18-10
Interrupts 18-11

USB OTG Controller Programming Model....18-13
Enabling SPRAM ECCs....18-13
Host Operation....18-13
Device Operation....18-15
USB 2.0 OTG Controller Address Map and Register Definitions....18-16
usb_globgrp Address Map....18-17
usb_hostgrp Address Map....18-149
usb_devgrp Address Map....18-611
usb_pwrclkgrp Address Map....18-1393
usb_DWC_otg_DFIFO Address Map....18-1396
usb_DWC_otg_DFIFO_Direct_access Address Map....18-1398
Document Revision History....18-1398

SPI Controller....19-1
Features of the SPI Controller....19-1
SPI Block Diagram and System Integration....19-2
SPI Block Diagram....19-2
SPI Controller Signal Description....19-3
Interface to HPS I/O....19-3
FPGA Routing....19-3
Functional Description of the SPI Controller....19-5
Protocol Details and Standards Compliance....19-5
SPI Controller Overview....19-5
Transfer Modes....19-8
SPI Master....19-9
SPI Slave....19-12
Partner Connection Interfaces....19-16
DMA Controller Interface....19-21
Slave Interface....19-21
Clocks and Resets....19-22
SPI Programming Model....19-23
Master SPI and SSP Serial Transfers....19-24
Master Microwire Serial Transfers....19-26
Slave SPI and SSP Serial Transfers....19-28
Slave Microwire Serial Transfers....19-29
Software Control for Slave Selection....19-29
DMA Controller Operation....19-30
SPI Controller Address Map and Register Definitions....19-33
spim Address Map....19-33
spis Address Map....19-88
Document Revision History....19-132

I^2C Controller....20-1
Features of the I^2C Controller....20-1
I^2C Controller Block Diagram and System Integration....20-2
I^2C Controller Signal Description....20-3
Functional Description of the I^2C Controller....20-5

Feature Usage....20-5
Behavior....20-5
Protocol Details....20-7
Multiple Master Arbitration....20-11
Clock Frequency Configuration....20-13
SDA Hold Time....20-15
DMA Controller Interface....20-15
Clocks....20-15
Resets....20-15
I2C Controller Programming Model....20-16
Slave Mode Operation....20-16
Master Mode Operation....20-20
Disabling the I2C Controller....20-23
Abort Transfer....20-23
DMA Controller Operation....20-23
I 2 C Controller Address Map and Register Definitions....20-27
i2c Address Map....20-27
i2c_emac Address Map....20-121
Document Revision History....20-239

UART Controller....21-1
UART Controller Features....21-1
UART Controller Block Diagram and System Integration....21-2
UART Controller Signal Description....21-3
HPS I/O Pins....21-3
FPGA Routing....21-4
Functional Description of the UART Controller....21-4
FIFO Buffer Support....21-4
UART(RS232) Serial Protocol....21-5
Automatic Flow Control....21-6
Clocks....21-7
Resets....21-8
Interrupts....21-8
DMA Controller Operation....21-11
Transmit FIFO Underflow....21-12
Transmit Watermark Level....21-12
Transmit FIFO Overflow....21-13
Receive FIFO Overflow....21-14
Receive Watermark Level....21-14
Receive FIFO Underflow....21-14
UART Controller Address Map and Register Definitions....21-15
uart Address Map....21-15
Document Revision History....21-109

General-Purpose I/O Interface....22-1
Features of the GPIO Interface....22-1
GPIO Interface Block Diagram and System Integration....22-2

Functional Description of the GPIO Interface....22-3
Debounce Operation....22-3
Pin Directions....22-3
Taking the GPIO Interface Out of Reset....22-3
GPIO Interface Programming Model....22-4
General-Purpose I/O Interface Address Map and Register Definitions....22-4
gpio Address Map....22-4
Document Revision History....22-37

Timer....23-1
Features of the Timer....23-1
Timer Block Diagram and System Integration....23-1
Functional Description of the Timer....23-2
Clocks....23-3
Resets....23-3
Interrupts....23-3
Timer Programming Model....23-4
Initialization....23-4
Enabling the Timer....23-4
Disabling the Timer....23-4
Loading the Timer Countdown Value....23-4
Servicing Interrupts....23-5
Timer Address Map and Register Definitions....23-5
timer_sys Address Map....23-5
timer Address Map....23-20
Document Revision History....23-35

Watchdog Timer....24-1
Features of the Watchdog Timer....24-1
Watchdog Timer Block Diagram and System Integration....24-2
Functional Description of the Watchdog Timer....24-2
Watchdog Timer Counter....24-2
Watchdog Timer Pause Mode....24-3
Watchdog Timer Clocks....24-3
Watchdog Timer Resets....24-4
Watchdog Timer Programming Model....24-4
Setting the Timeout Period Values....24-4
Selecting the Output Response Mode....24-5
Enabling and Initially Starting a Watchdog Timer....24-5
Reloading a Watchdog Counter....24-5
Pausing a Watchdog Timer....24-5
Disabling and Stopping a Watchdog Timer....24-5
Watchdog Timer State Machine....24-5
Watchdog Timer Address Map and Register Definitions....24-7
watchdog Address Map....24-7
Document Revision History....24-27

Hard Processor System I/O Pin Multiplexing.....25-1
Features of the HPS I/O Block..... 25-1
HPS I/O Block Diagram and System Integration.....25-2
Functional Description of the HPS I/O..... 25-3
Dedicated I/O Pins.....25-3
Shared I/O Pins..... 25-4
FPGA Access.....25-5
Control Registers.....25-5
Configuring HPS I/O Multiplexing.....25-8
Test Considerations..... 25-9
I/O Pin MUX Address Map and Register Definitions for Arria 10.....25-9
Document Revision History.....25-10

Introduction to the HPS Component..... 26-1
MPU Subsystem..... 26-2
ARM CoreSight Debug Components.....26-2
Interconnect.....26-2
HPS-to-FPGA Interfaces.....26-2
Memory Controllers..... 26-3
HPS SDRAM Controller..... 26-3
Support Peripherals.....26-5
Interface Peripherals..... 26-5
On-Chip Memories.....26-6
Document Revision History.....26-6

Instantiating the HPS Component..... 27-1
Using the HPS Parameter Editor..... 27-1
FPGA Interfaces..... 27-2
General Interfaces..... 27-3
FPGA-to-HPS SDRAM Interface.....27-6
DMA Peripheral Request..... 27-7
Security Manager..... 27-7
Interrupts.....27-7
AXI Bridges.....27-10
Configuring HPS Clocks and Resets..... 27-11
Alternate Clock Source from FPGA.....27-11
User Clocks..... 27-11
Reset Interfaces.....27-12
Peripheral FPGA Clocks..... 27-13
Configuring Peripheral Pin Multiplexing.....27-14
Configuring Peripherals.....27-14
Connecting Unassigned Pins to GPIO.....27-16
Peripheral Signals Routed to FPGA 27-16
Configuring the External Memory Interface.....27-16
Using the Address Span Extender Component.....27-17

Generating and Compiling the HPS Component 27-18
Document Revision History 27-18

HPS Component Interfaces 28-1
Memory-Mapped Interfaces 28-1
FPGA-to-HPS Bridge 28-1
HPS-to-FPGA and Lightweight HPS-to-FPGA Bridges 28-2
FPGA-to-HPS SDRAM Interface 28-3
EMIF Conduit 28-4
Clocks 28-4
Alternative Clock Inputs to HPS PLLs 28-4
User Clocks 28-4
AXI Bridge FPGA Interface Clocks 28-4
SDRAM Clocks 28-4
Peripheral FPGA Clocks 28-5
Resets 28-6
HPS-to-FPGA Reset Interfaces 28-6
HPS External Reset Request 28-6
Peripheral Reset Interfaces 28-6
Debug and Trace Interfaces 28-6
FPGA System Trace Macrocell Events Interface 28-6
FPGA Cross Trigger Interface 28-6
Debug APB Interface 28-7
Peripheral Signal Interfaces 28-7
Qsys Peripheral Port Interface Mapping 28-7
DMA Controller Interface 28-14
Other Interfaces 28-15
MPU Standby and Event Interfaces 28-15
General Purpose Signals 28-16
FPGA-to-HPS Interrupts 28-16
Boot from FPGA Interface 28-16
Security Manager 28-16
Document Revision History 28-17

Simulating the HPS Component 29-1
Simulation Flows 29-2
Setting Up the HPS Component for Simulation 29-3
Generating the HPS Simulation Model in Qsys 29-5
Running the Simulations 29-5
Clock and Reset Interfaces 29-9
Clock Interface 29-9
Reset Interface 29-10
FPGA-to-HPS AXI Slave Interface 29-11
HPS-to-FPGA AXI Master Interface 29-11
Lightweight HPS-to-FPGA AXI Master Interface 29-12
HPS-to-FPGA MPU Event Interface 29-13
Interrupts Interface 29-13

HPS-to-FPGA Debug APB Interface 29-14
FPGA-to-HPS System Trace Macrocell Hardware Event Interface 29-15
HPS-to-FPGA Cross-Trigger Interface 29-15
FPGA-to-HPS DMA Handshake Interface 29-16
Boot from FPGA Interface 29-18
Security Manager Anti-Tamper Signals Interface 29-18
EMIF Conduit 29-18
Pin MUX and Peripherals 29-19
HPS Conduit Interfaces Connecting to the HPS I/O 29-19
HPS Conduit Interfaces Connecting to the FPGA 29-28
Document Revision History 29-31

Booting and Configuration A-1
Boot Overview A-1
FPGA Configuration Overview A-2
Booting and Configuration Options A-2
Boot Definitions A-5
Reset A-5
Boot ROM A-6
Boot Select A-6
Flash Memory Devices for Booting A-13
Clock Select A-25
I/O Configuration A-28
L4 Watchdog 0 Timer A-28
Second-Stage Boot Loader A-29
Secure Boot A-30
Boot ROM Flow A-31
Second-Stage Boot Flow A-35
Typical Boot Flow (Non-Secure) A-35
Secure Boot Flow A-38
HPS State on Entry to the Second-Stage Boot Loader A-43
Loading the Second-Stage Boot Loader Image A-44
FPGA Configuration A-46
Full FPGA Configuration Flow Through HPS A-47
Early I/O Release FPGA Configuration Flow Through HPS A-48
Arria 10 SoC FPGA Configuration Sequence Through FPGA Manager A-50
FPGA Reconfiguration A-51
Full FPGA Reconfiguration A-51
Arria 10 SoC FPGA Partial Reconfiguration Sequence Through FPGA Manager A-54
Document Revision History A-56

Introduction to the Hard Processor System

1

2016.10.28

a10_5v4 Subscribe Send Feedback

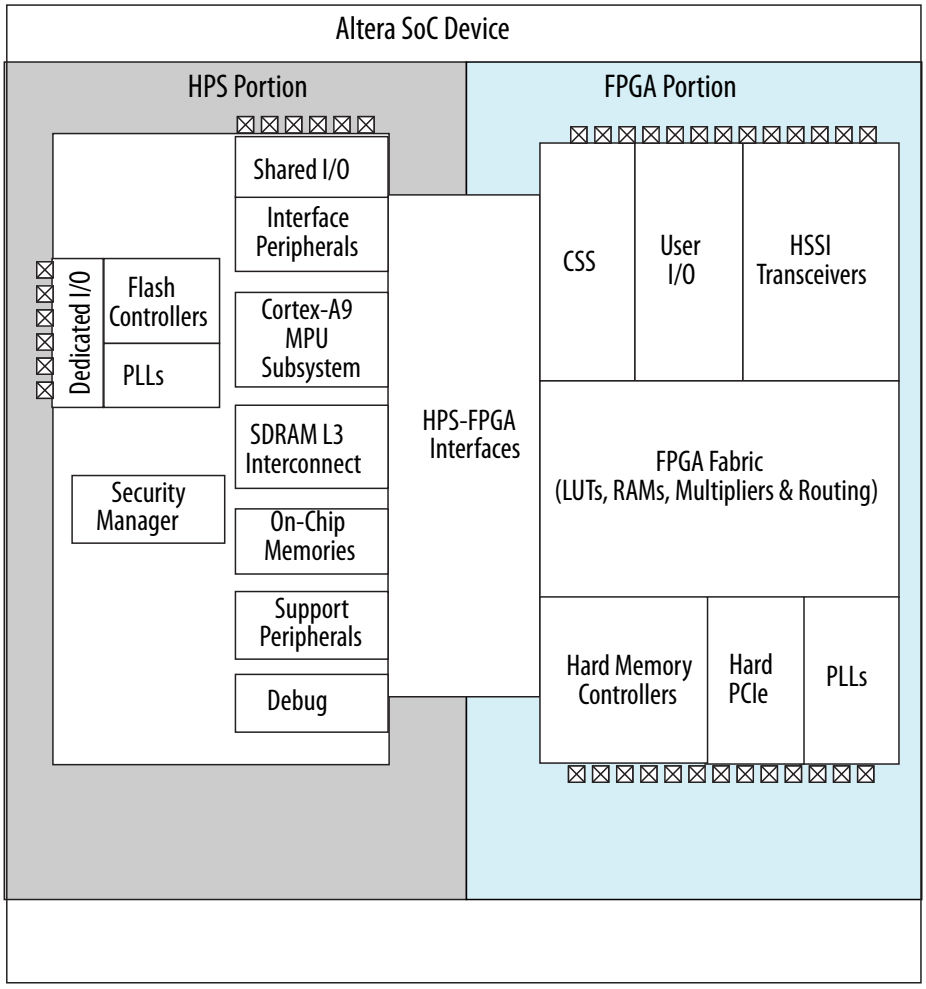
The Arria® 10 system-on-a-chip (SoC) is composed of two distinct portions- a dual-core ARM® Cortex™-A9 hard processor system (HPS) and an FPGA. The HPS architecture integrates a wide set of peripherals that reduce board size and increase performance within a system. A dedicated security manager within the HPS supports secure boot with the ability to authenticate, decrypt and provide tamper event response.

Figure 1-1 shows a high-level block diagram of the Altera® Arria 10 SoC device. Blocks connected to device pins have symbols (square with an X) adjacent to them in the figure.

The SoC features the following types of I/O pins:

- Dedicated I/O - I/O that is dedicated to an external non-volatile storage device (for boot ROM), HPS clock and resets.
- Shared I/O - I/O that can be assigned to peripherals in the HPS or FPGA logic.
- FPGA I/O - I/O that is dedicated to the FPGA fabric.

Figure 1-1: Altera SoC Device Block Diagram



The HPS consists of the following types of modules:

- Microprocessor unit (MPU) subsystem with a dual ARM Cortex-A9 MPCore® processors
- Flash memory controllers
- SDRAM L3 interconnect
- System interconnect
- On-chip memories
- Support peripherals
- Interface peripherals
- Debug components
- Phase-locked loops (PLLs)

The HPS incorporates third-party intellectual property (IP) from several vendors.

The dual-processor HPS supports symmetric (SMP) and asymmetric (AMP) multiprocessing.

The FPGA portion of the device contains:

- FPGA fabric
- Configuration sub-system (CSS)
- PLLs
- High-speed serial interface (HSSI) transceivers, depending on the device variant
- Hard PCI Express® (PCI-e) controllers
- Hard memory controllers

The HPS and FPGA communicate with each other through bus interfaces that bridge the two distinct portions. On a power-on reset, the HPS can boot from multiple sources, including the FPGA fabric and external flash. The FPGA can be configured through the HPS or an externally supported device.

The HPS and FPGA portions of the device each have their own pins. The HPS has dedicated I/O pins and can also share some FPGA I/O pins. Pin assignments are configured when the HPS component is instantiated in Qsys. At boot time, software executing on the HPS assigns the I/O pins to the available HPS modules and configures the I/O pins through I/O control registers. For more information, refer to the "Hard Processor System I/O Pin Multiplexing" chapter. The FPGA I/O pins are configured by an FPGA configuration image through the HPS or any external source supported by the device.

The FPGA fabric and the HPS must be powered at the same time. Once powered, the FPGA fabric and the HPS can be configured independently thus providing you with more design flexibility:

- You can boot the HPS independently. After the HPS is running, the HPS can fully or partially reconfigure the FPGA fabric at any time under software control. The HPS can also configure other FPGAs on the board through the FPGA configuration controller.
- You can configure the FPGA fabric first, and then boot the HPS from the memory accessible to the FPGA fabric.

Related Information

- **Hard Processor System I/O Pin Multiplexing** on page 25-1
  Information about I/O pin configuration. The FPGA I/O pins are configured by an FPGA configuration image through the HPS or any external source supported by the device.
- **Arria 10 Device Datasheet**

Features of the HPS

The main modules of the HPS are:

- MPU subsystem featuring a dual-core ARM Cortex-A9 MPCore processor
- System interconnect that includes three memory-mapped interfaces between the HPS and FPGA:
  - HPS-to-FPGA: 32-, 64-, or 128-bit wide AXI-4
  - Lightweight HPS-to-FPGA: 32-bit wide AXI-4
  - FPGA-to-HPS: 32-, 64-, or 128-bit wide ACE
- General-purpose direct memory access (DMA) controller
- Security manager
- Supports peripheral memories with single-error correction and double-error detection (SECDED)
- Three Ethernet media access controllers (EMACs)
- Two USB 2.0 on-the-go (OTG) controllers
- NAND flash controller
- Quad serial peripheral interface (QSPI) flash controller
- Secure digital/multimedia card (SD/MMC) controller
- Two serial peripheral interface (SPI) master controllers

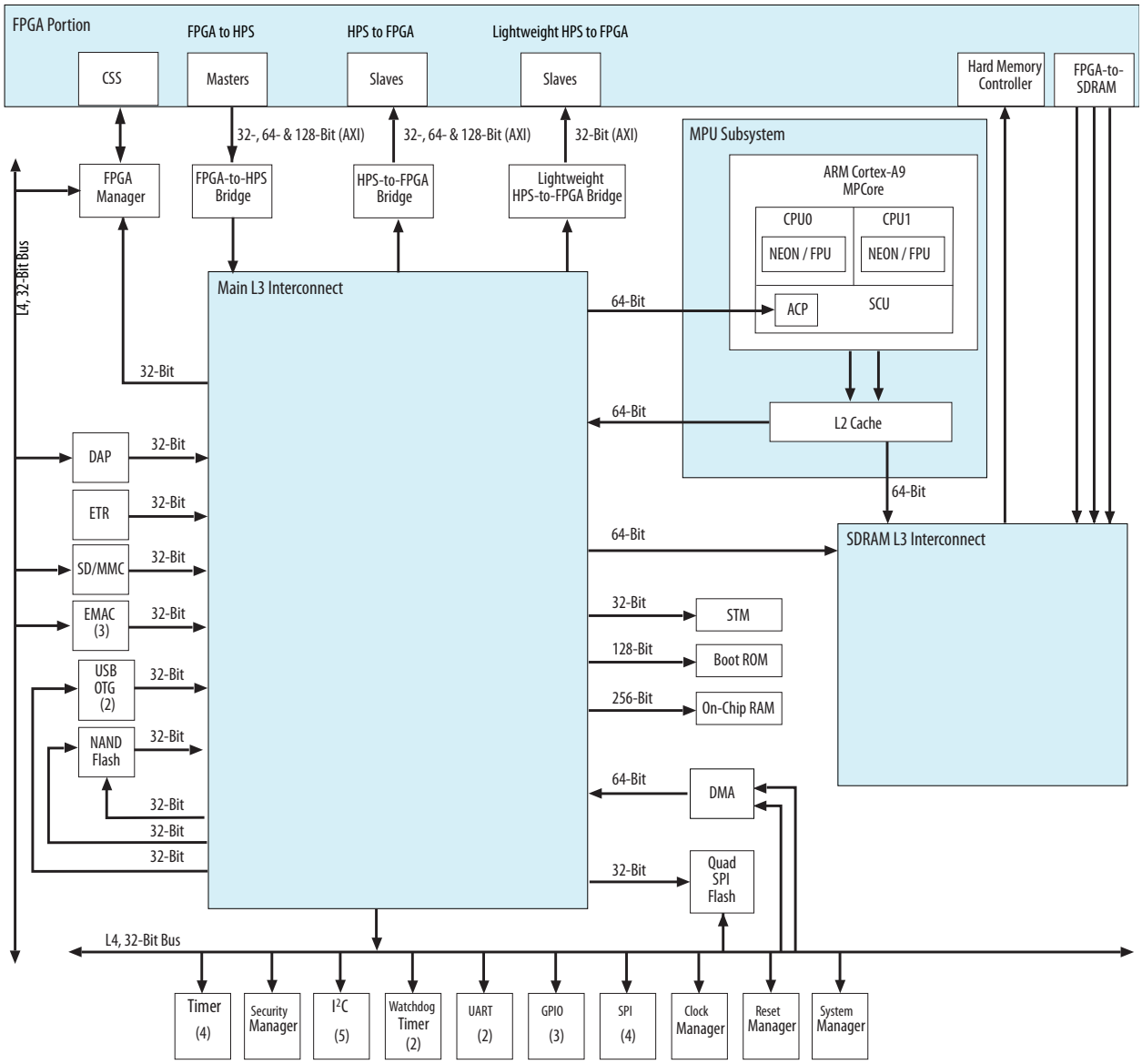
- Two SPI slave controllers
- Five inter-integrated circuit (I^2C) controllers(1)
- 256 KB on-chip RAM
- 128 KB on-chip boot ROM
- Two UARTs
- Four system timers
- Two watchdog timers
- Three general-purpose I/O (GPIO) interfaces
- ARM Coresight™ debug components:
  - Debug access port (DAP)
  - Trace port interface unit (TPIU)
  - System trace macrocell (STM)
  - Program trace macrocell (PTM)
  - Embedded trace router (ETR)
  - Embedded cross trigger (ECT)
- System manager
- Clock manager
- Reset manager
- FPGA manager

(1) Three of the five I^2Cs, can optionally be configured to provide PHY management support for each EMAC controller.

HPS Block Diagram and System Integration

HPS Block Diagram

Figure 1-2: HPS Block Diagram



Cortex-A9 MPCore

The MPU subsystem is a stand-alone, full-featured ARM Cortex-A9 MPCore dual-core 32-bit application processor. It provides the following functionality:

- ARM Cortex-A9 MPCore
  - Two ARM Cortex-A9 processors
  - NEON™ single instruction, multiple data (SIMD) coprocessor and vector floating-point v3 (VFPv3) per processor
  - Snoop control unit (SCU) to ensure coherency between processors
  - Accelerator coherency port (ACP) that accepts coherency memory access requests
  - Interrupt controller
  - One general-purpose timer and one watchdog timer per processor
  - Debug and trace features
  - 32 KB instruction and 32 KB data level 1 (L1) caches per processor
  - Memory management unit (MMU) per processor
- ARM L2-310 level 2 (L2) cache
  - Shared 512 KB L2 cache

As shown in the "HPS Block Diagram", the L2 cache has one 64-bit master port that is connected to the main L3 interconnect and one 64-bit master port connected to the SDRAM L3 interconnect. A programmable address filter in the L2 cache controls which portions of the 32-bit physical address space can be accessed by each master.

Related Information

HPS Block Diagram on page 1-5

HPS Interfaces

The Arria 10 device family provides multiple communication channels to the HPS.

HPS–FPGA Memory-Mapped Interfaces

The HPS–FPGA memory-mapped interfaces provide the major communication channels between the HPS and the FPGA fabric. The HPS–FPGA memory-mapped interfaces include:

- FPGA–to–HPS bridge—a high–performance bus with a configurable data width of 32, 64, or 128 bits, allowing the FPGA fabric to master transactions to the slaves in the HPS. This interface allows the FPGA fabric to have full visibility into the HPS address space. This interface also provides access to the coherent memory interface
- HPS–to–FPGA bridge—a high–performance interface with a configurable data width of 32, 64, or 128 bits, allowing the HPS to master transactions to slaves in the FPGA fabric
- Lightweight HPS–to–FPGA bridge—an interface with a 32–bit fixed data width, allowing the HPS to master transactions to slaves in the FPGA fabric

Related Information

HPS-FPGA Bridges on page 8-1

Other HPS Interfaces

- TPIU trace—sends trace data created in the HPS to the FPGA fabric
- FPGA System Trace Macrocell (STM)—an interface that allows the FPGA fabric to send hardware events to be stored in the HPS trace data
- FPGA cross–trigger—an interface that allows the CoreSight trigger system to send triggers to IP cores in the FPGA, and vise versa
- DMA peripheral interface—multiple peripheral–request channels
- FPGA manager interface—signals that communicate with the FPGA fabric for boot and configuration
- Interrupts—allow soft IP cores to supply interrupts directly to the MPU interrupt controller
- MPU standby and events—signals that notify the FPGA fabric that the MPU is in standby mode and signals that wake up Cortex–A9 processors from a wait for event (WFE) state
- HPS debug interface – an interface that allows the HPS debug control domain (debug APB) to extend into FPGA

Another HPS–FPGA communications channel is FPGA clocks and resets.

System Interconnect

The system interconnect consists of the main L3 interconnect, SDRAM L3 interconnect, and level 4 (L4) buses. The system interconnect is based on the Arteris™ FlexNoC® network-on-chip (NoC) interconnect module. The system interconnect incorporates configurable secure firewalls protecting each peripheral.

The L4 buses are each connected to a master in the main L3 interconnect. Each L4 bus is 32 bits wide and is connected to multiple slaves. Each L4 bus operates on a separate clock source.

The SDRAM L3 interconnect consists of the SDRAM adapter and the SDRAM scheduler. SDRAM is protected by firewalls in the SDRAM L3 interconnect.

Related Information

- **System Interconnect** on page 7-1
- **Arria 10 HPS Secure Firewalls** on page 6-24

You can use the system interconnect firewalls to enforce security policies for slave and memory regions in the system interconnect.

Arria 10 HPS SDRAM L3 Interconnect

The SDRAM L3 interconnect is part of the system interconnect and connects the HPS to the hard memory controller that is located in the FPGA fabric. The SDRAM L3 interconnect is composed of the SDRAM adapter and the SDRAM scheduler, which are secured by firewalls. It supports ARM Advanced Microcontroller Bus Architecture (AMBA®) Advanced eXtensible Interface (AXI™) quality of service (QoS) for the fabric interfaces

The hard memory controller implements the following high-level features:

- Support for double data rate 3 (DDR3) and DDR4 devices
- Software-configurable priority scheduling on individual SDRAM bursts
- Error correction code (ECC) support, including calculation, single-bit error correction and write-back, and error counters
- Fully-programmable timing parameter support for all JEDEC-specified timing parameters
- All ports support memory protection and mutual-exclusive accesses
- FPGA-to-SDRAM interface—a configurable interface to the SDRAM scheduler in the SDRAM L3 interconnect You can configure the following parameters:
  - Up to three bridges
  - Up to 288 bits across all three ports

Related Information

SDRAM L3 Interconnect Block Diagram and System Integration on page 7-9
For more information on the components of the SDRAM L3 Interconnect and how they interface to the hard memory controller in the FPGA, refer to this diagram.

SDRAM Adapter

The SDRAM adapter is responsible for bridging the hard memory controller in the FPGA fabric to the SDRAM scheduler. The adapter is also responsible for error correction code (ECC) generation and checking.

SDRAM Scheduler

The SDRAM scheduler accepts read and write requests from the processor, HPS peripheral masters, and soft logic in FPGA through the FPGA-to-SDRAM interface. It is programmed with memory timings, allowing it to opimize memory access.

On-Chip Memory

On-Chip RAM

The on-chip RAM offers the following features:

- 256 KB size
- 64-bit slave interface
- High performance for all burst lengths
- ECC support provides detection of single–bit and double–bit errors and correction for single-bit errors
- Can clear the on-chip RAM on reset

Related Information

On-Chip Memory on page 12-1

Boot ROM

The boot ROM offers the following features:

- 32-bit interface
- Capable of data transfers for every clock cycle
- 128 KB size
- Contains the code required to support both secure and non-secure HPS boot from cold or warm reset
- Used exclusively for booting the HPS

Related Information

- **On-Chip Memory** on page 12-1
- **Booting and Configuration** on page 30-1

Flash Memory Controllers

NAND Flash Controller

The NAND flash controller is based on the Cadence® Design IP® NAND Flash Memory Controller and offers the following functionality and features:

- Supports up to four chip selects. One chip select is connected to an HPS I/O pin and the rest can be connected to the FPGA I/O pins.
- Integrated descriptor-based DMA controller
- 8-bit and 16-bit ONFI 1.0 NAND flash devices
- Programmable page sizes of 512 bytes, 2 KB, 4 KB, and 8 KB
- Supports 32, 64, 128, 256, 384, and 512 pages per block
- Programmable hardware ECC for single-level cell (SLC) and multi-level cell (MLC) devices
- 512-byte ECC sector size with 4-, 8-, or 16-bit correction
- 1 KB ECC sector size with 24-bit correction

Related Information

NAND Flash Controller on page 13-1

Quad SPI Flash Controller

The quad SPI flash controller is used for access to serial NOR flash devices. It supports standard SPI flash devices as well as high-performance dual and quad SPI flash devices. The Quad SPI flash controller is based on the Cadence Quad SPI Flash Controller and offers the following features:

- Supports SPIx1, SPIx2, or SPIx4 (Quad SPI) serial NOR flash devices
- Supports direct access and indirect access modes
- Supports single, dual, and quad I/O instructions
- Support up to four chip selects
- Programmable write-protected regions
- Programmable delays between transactions
- Programmable device sizes
- Support eXecute-In-Place (XIP) mode
- Programmable baud rate generator to generate device clocks

Related Information

Quad SPI Flash Controller on page 15-1

SD/MMC Controller

The Secure Digital (SD), Multimedia Card (MMC), (SD/MMC) and CE-ATA host controller is based on the Synopsys® DesignWare® Mobile Storage Host controller and offers the following features:

- Integrated descriptor-based DMA
- Supports CE-ATA digital protocol commands
- Supports single card
- Single data rate (SDR) mode only
- Programmable card width: 1-, 4-, and 8-bit
- Programmable card types: SD, SDIO, or MMC
- Up to 64 KB programmable block size
- Supports the following standards and card types:
  - SD, including eSD—version 3.0[(2)]
  - SDIO, including embedded SDIO (eSDIO)—version 3.0[(3)]
  - CE-ATA—version 1.1
- Supports various types of multimedia cards, MMC version 4.41[(4)]
  - MMC: 1-bit data bus
  - Reduced-size MMC (RSMMC): 1-bit and 4-bit data bus
  - MMCMobile: 1-bit data bus
- Supports embedded MMC (eMMC) version 4.5[(5)]
  - 1-bit, 4-bit, and 8-bit data bus

Note: For an inclusive list of the programmable card types and versions supported, refer to the *SD/MMC Controller* chapter.

Related Information

SD/MMC Controller on page 14-1

Support Peripherals

Clock Manager

The clock manager is responsible for providing software-programmable clock control to configure all clocks generated in the HPS. The clock manager offers the following features:

- Manages clocks for HPS
- Supports clock gating at the signal level
- Supports dynamic clock tuning

Related Information

Clock Manager on page 2-1

(2) Does not support SDR50, SDR104, and DDR50 modes.
(3) Does not support SDR50, SDR104, and DDR50 modes.
(4) Does not support DDR mode.
(5) Does not support DDR and HS200 mode.

Reset Manager

The reset domains and sequences support several security features. The security manager brings the reset manager out of reset only when device security is confirmed. Afterwards, the reset manager brings the rest of the HPS system out of reset. The reset manager performs the following functions:

- Manages resets for HPS
- Controls the resets for cold, warm, debug, and TAP reset domains
- Controls the RAM clearing domain separately

Related Information

- **Reset Manager** on page 3-1
- **SoC Security** on page 6-1

Security Manager

The security manager module is integrated in the HPS and provides the overall management of security within the SoC, including:

- Recognition of secure fuse configuration, preventing non-secure device startup
- State control and check of the security features
- Secure boot options
- Secure debug options
- Anti-tamper support

Related Information

SoC Security on page 6-1

System Manager

The system manager contains logic to control system functions and logic to control other modules that need external control signals provided as part of system integration. The system manager offers the following features:

- Provides combined ECC status and interrupts from other HPS modules with ECC-protected RAM
- Low-level control of peripheral features not accessible through the control and status registers (CSRs)

Related Information

System Manager on page 5-1

System Timers

The HPS provides four 32-bit general-purpose timers connected to the level 4 (L4) peripheral bus. The four system timers are based on the Synopsys DesignWare Advanced Peripheral Bus (APB) Timers peripheral and offer the following features:

- 32-bit timer resolution
- Free-running timer mode
- Programmable time-out period up to approximately 86 seconds (assuming a 50 MHz input clock frequency)
- Interrupt generation

Related Information

Timer on page 23-1

System Watchdog Timers

The two system watchdog timers are based on the Synopsys DesignWare APB Watchdog Timer peripheral and offer the following features:

- 32-bit timer resolution
- Interrupt request
- Reset request
- Programmable time-out period up to approximately 86 seconds (assuming a 50 MHz input clock frequency)

Note: Countdown can be paused when the MPU is in debug mode.

Related Information

Watchdog Timer on page 24-1

DMA Controller

The DMA controller provides high-bandwidth data transfers for modules without integrated DMA controllers. The DMA controller is based on the ARM Corelink™ DMA Controller (DMA-330) and offers the following features:

- Micro-coded to support flexible transfer types
  - Memory-to-memory
  - Memory-to-peripheral
  - Peripheral-to-memory
  - Scatter-gather
- Supports up to eight channels
- Supports flow control with 32 peripheral handshake interfaces

Related Information

DMA Controller on page 16-1

FPGA Manager

The FPGA manager manages and monitors the FPGA portion of the SoC device. The FPGA manager can configure the FPGA fabric from the HPS, monitor the state of the FPGA, and drive or sample signals to or from the FPGA fabric.

Related Information

FPGA Manager on page 4-1

Error Checking and Correction Controller

ECC controllers provide single- and double-bit error memory protection for integrated on-chip RAM and peripheral RAMs within the HPS.

The following peripherals have integrated ECC-protected memories:

- USB OTG 0 - 1
- SD/MMC Controller
- EMAC 0 - 2
- DMA controller
- NAND flash controller
- QSPI flash controller

Features of the ECC controller:

- Single-bit error detection and correction
- Double-bit error detection
- Interrupts generated on single- and double-bit errors

Related Information

Error Checking and Correction Controller on page 11-1
Error Checking and Correction (ECC) controllers provide single- and double-bit error memory protection for integrated on-chip RAM and peripheral RAMs within the hard processor system (HPS).

Interface Peripherals

EMACs

The three EMACs are based on the Synopsys DesignWare 3504-0 Universal 10/100/1000 Ethernet MAC and offer the following features:

- Supports 10, 100, and 1000 Mbps standard
- PHY interfaces supported through the HPS I/O pins:
  - Reduced media independent interface (RMII) and Reduced gigabit media independent interface (RGMII) through the HPS I/O pins
- PHY interfaces supported using adapter logic to route signals to the FPGA I/O pins:
  - Media independent interface (MII), Gigabit media independent interface (GMII), RMII, RGMII, and Serial gigabit media independent interface (SGMII) (with external conversion logic) through the FPGA pins
- Integrated DMA controller
- Supports IEEE 1588-2002 and IEEE 1588-2008 standards for precision networked clock synchronization
- IEEE 802.3-az, version D2.0 of Energy Efficient Ethernet
- Supports IEEE 802.1Q Virtual local area network (VLAN) tag detection for reception frames
- PHY Management control through Management data input/output (MDIO) interface or I^2C interface
- 4 KB TX FIFO and 16 KB RX FIFO RAM
- Supports a variety of address filtering modes

Related Information

Ethernet Media Access Controller on page 17-1

USB Controllers

The HPS provides two USB 2.0 Hi-Speed On-the-Go (OTG) controllers from Synopsys DesignWare. The USB controller signals cannot be routed to the FPGA like those of other peripherals; instead they are routed to the dedicated I/O.

Each of the USB controllers offers the following features:

- Complies with the following specifications:
  - USB OTG Revision 1.3
  - USB OTG Revision 2.0
  - Embedded Host Supplement to the USB Revision 2.0 Specification
- Supports software-configurable modes of operation between OTG 1.3 and OTG 2.0
- Supports all USB 2.0 speeds:
  - High speed (HS, 480-Mbps)
  - Full speed (FS, 12-Mbps)
  - Low speed (LS, 1.5-Mbps)

  Note: In host mode, all speeds are supported; however, in device mode, only high speed and full speed are supported.
- Local buffering with Error Correction Code (ECC) support

  Note: The USB 2.0 OTG controller does not support the following interface standards:
  - Enhanced Host Controller Interface (EHCI)
  - Open Host Controller Interface (OHCI)
  - Universal Host Controller Interface (UHCI)
- Supports USB 2.0 Transceiver Macrocell Interface Plus (UTMI+) Low Pin Interface (ULPI) PHYs (SDR mode only)
- Supports up to 16 bidirectional endpoints, including control endpoint 0

  Note: Only seven periodic device IN endpoints are supported.
- Supports up to 16 host channels

  Note: In host mode, when the number of device endpoints is greater than the number of host channels, software can reprogram the channels to support up to 127 devices, each having 32 endpoints (IN + OUT), for a maximum of 4,064 endpoints.
- Supports generic root hub
- Supports automatic ping capability

Related Information

- **USB 2.0 OTG Controller** on page 18-1
- **Universal Serial Bus (USB) website**
  Additional information is available in the On The Go and Embedded Host Supplement to the USB Revision 2.0 Specification, which you can download from the USB Implementers Forum website.

I²C Controllers

The five I²C controllers are based on Synopsys DesignWare APB I²C controller which offer the following features:

- Three controllers optionally can be used as a control interface for Ethernet PHY communication

  Note: When an I²C controller is used for Ethernet, it will take the place of the MDIO pins.
- Support both 100 KBps and 400 KBps modes
- Support both 7-bit and 10-bit addressing modes
- Support master and slave operating mode
- Direct access for host processor
- DMA controller may be used for large transfers

Related Information

I2C Controller on page 20-1

UARTs

The HPS provides two UART controllers to provide asynchronous serial communications. The two UART modules are based on Synopsys DesignWare APB Universal Asynchronous Receiver/ Transmitter peripheral and offer the following features:

- 16550-compatible UART
- Support automatic flow control as specified in 16750 standard
- Programmable baud rate up to 6.25 MBaud (with 100MHz reference clock)
- Direct access for host processor
- DMA controller may be used for large transfers
- 128-byte transmit and receive FIFO buffers
- Separate thresholds for DMA request and handshake signals to maximize throughput

Related Information

UART Controller on page 21-1

SPI Master Controllers

The two SPI master controllers are based on Synopsys DesignWare Synchronous Serial Interface (SSI) controller and has a maximum bit rate of 60Mbps. The following features are offered:

- Programmable data frame size from 4 bits to 16 bits
- Supports full- and half-duplex modes
- Supports two chip selects connected to HPS I/O
- Supports four chip selects connected to the FPGA fabric
- Direct access for host processor
- DMA controller may be used for large transfers
- Programmable master serial bit rate
- Support for `rxd` sample delay
- Transmit and receive FIFO buffers are 256 words deep
- Choice of Motorola® SPI, Texas Instruments® Synchronous Serial Protocol or National Semiconductor® Microwire protocol

Related Information

SPI Controller on page 19-1

SPI Slave Controllers

The two SPI slave controllers are based on Synopsys DesignWare Synchronous Serial Interface (SSI) controller and has a maximum bit rate of 50 Mbps. The following features are offered:

- Programmable data frame size from 4 bits to 16 bits
- Supports full- and half-duplex modes
- Direct access for host processor
- DMA controller may be used for large transfers
- Transmit and receive FIFO buffers are 256 words deep

Related Information

SPI Controller on page 19-1

GPIO Interfaces

The HPS provides three GPIO interfaces that are based on Synopsys DesignWare APB General Purpose Programming I/O peripheral and offer the following features:

- Supports digital de-bounce
- Configurable interrupt mode
- Supports up to 17 dedicated 1.8 V and 3.0 V HPS I/O pins used for clock, reset, and external flash devices
- Supports up to 48 shared 3.0 V I/O pins that can be used by either the HPS or the FPGA. These pins are useful for Ethernet, USB, and other communication functions

Related Information

General-Purpose I/O Interface on page 22-1

CoreSight Debug and Trace

The CoreSight debug and trace system offers the following features:

- Real-time program flow instruction trace through a separate PTM for each processor
- Host debugger JTAG interface
- Connections for cross-trigger and STM-to-FPGA interfaces, which enable soft IP cores to generate of triggers and system trace messages
- Custom message injection through STM into trace stream for delivery to host debugger
- Capability to route trace data to any slave accessible to the ETR master, which is connected to the level 3 (L3) interconnect

Related Information

CoreSight Debug and Trace on page 10-1

Hard Processor System I/O Pin Multiplexing

The Stratix® 10 SoC has a total of 48 dedicated I/O pins that are used for hard processor system (HPS) operation, external flash memories, and external peripheral communication. A pin multiplexing mechanism allows the SoC to use the flexible I/O pins in a wide range of configurations.

Endian Support

The HPS is natively a little–endian system. All HPS slaves are little endian.

The processor masters are software configurable to interpret data as little endian, big endian, or byte–invariant (BE8). All other masters, including the USB 2.0 interface, are little endian. Registers in the MPU and L2 cache are little endian regardless of the endian mode of the CPUs.

Note: Altera strongly recommends that you only use little endian.

The FPGA–to–HPS, HPS–to–FPGA, FPGA–to–SDRAM, and lightweight HPS–to–FPGA interfaces are little endian.

If a processor is set to BE8 mode, software must convert endianness for accesses to peripherals and DMA linked lists in memory. The processor provides instructions to swap byte lanes for various sizes of data.

The ARM Cortex–A9 MPU supports a single instruction to change the endianness of the processor and provides the REV and REV16 instructions to swap the endianness of bytes or half–words respectively. The MMU page tables are software configurable to be organized as little–endian or BE8.

The ARM DMA controller is software configurable to perform byte lane swapping during a transfer.

Introduction to the Hard Processor System Address Map

The address map specifies the addresses of slaves, such as memory and peripherals, as viewed by the MPU and other masters. The HPS has multiple address spaces, defined in the following section.

Related Information

MPU Address Space on page 7-17

HPS Address Spaces

The following table shows the HPS address spaces and their sizes.

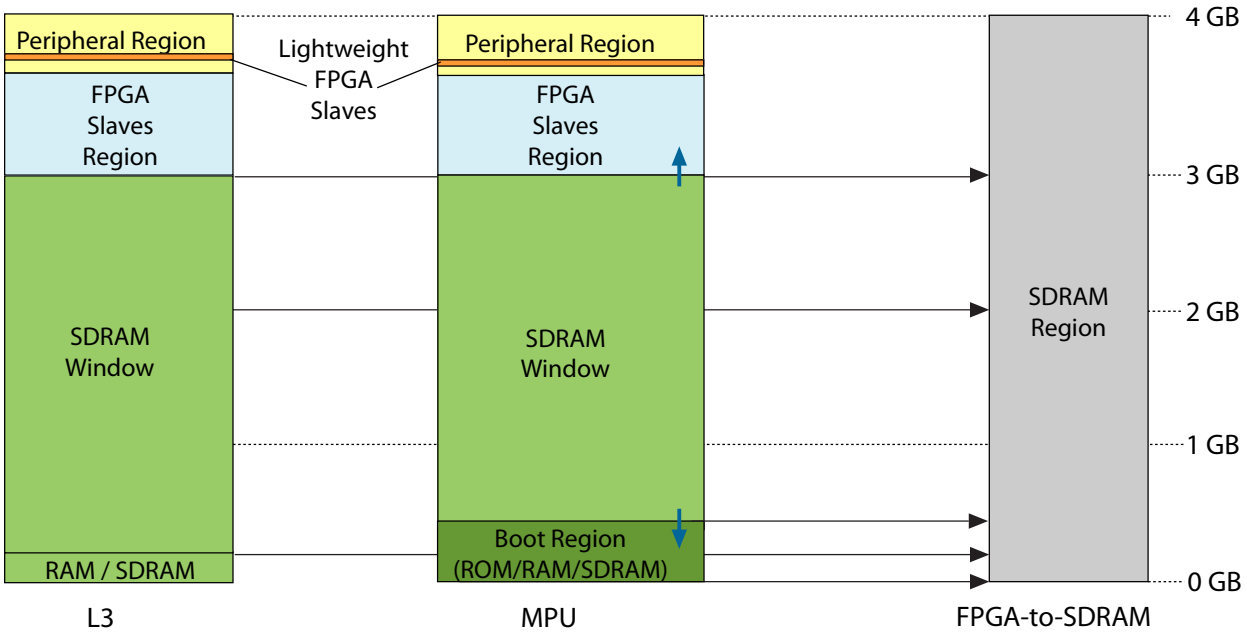
Table 1-1: HPS Address Spaces

Name	Description	Size
MPU	MPU subsystem	4 GB
L3	System interconnect	4 GB
SDRAM	SDRAM region	4 GB

Address spaces are divided into one or more nonoverlapping regions. For example, the MPU address space has the peripheral, FPGA slaves, SDRAM window, and boot regions.

The following figure shows the relationships between the HPS address spaces. The figure is not to scale.

Figure 1-3: HPS Address Space Relationships



The SDRAM window in the MPU address space can grow and shrink at the top and bottom (short, blue vertical arrows) at the expense of the FPGA slaves and boot regions. For specific details, refer to "MPU Address Space".

The following table shows the base address and size of each region that is common to the L3 and MPU address spaces.

Table 1-2: Common Address Space Regions

Region Name	Base Address	Size
FPGA slaves	0xC0000000	960 MB
Peripheral	0xFC000000	64 MB
Lightweight FPGA slaves	0xFF200000	2 MB

SDRAM Address Space

The SDRAM address space is up to 4 GB. The entire address space can be accessed through the FPGA-to-SDRAM interface from the FPGA fabric. The total amount of SDRAM addressable from the other address spaces can be configured at runtime.

There are cacheable and non-cacheable views into the SDRAM space. When a master of the L3 SDRAM interconnect performs a cacheable access to the SDRAM, the transaction is performed through the ACP port of the MPU subsystem. When a master of the SDRAM L3 interconnect performs a non-cacheable

access to the SDRAM, the transaction is performed through the 64-bit L3 interconnect master of the SDRAM L3 interconnect.

Related Information

System Interconnect on page 7-1
For more information about how to configure SDRAM address space.

MPU Address Space

The MPU address space is 4 GB and applies to addresses generated inside the MPU.

The MPU address space contains the following regions:

- The SDRAM window region provides access to a large, configurable portion of the 4 GB SDRAM address space.

The address filtering start and end registers in the L2 cache controller define the SDRAM window boundaries. The boundaries are megabyte-aligned. Addresses within the boundaries route to the SDRAM master. Addresses outside the boundaries route to the system interconnect master.

Related Information

- **HPS Address Spaces** on page 1-17
- **System Interconnect** on page 7-1
  For more information regarding SDRAM address mapping, refer to the System Interconnect chapter.
- **Cortex-A9 Microprocessor Unit Subsystem** on page 9-1
- **HPS Address Spaces** on page 1-17

FPGA Slaves Address Space

The FPGA Slaves address space is located in the FPGA core and accessed from the HPS through the HPS2FPGA AXI Bridge. The FPGA Slaves address space in the FPGA core is of unlimited size (soft logic in the FPGA performs address decoding). The L3 and MPU regions provide a window of nearly 1GB (actually 1GB less 64MB) into the FPGA Slaves address space. The base address of the FPGA Slaves Window is mapped to address 0x0 in the FPGA Slaves address space.

The HPS is able to boot from offset 0x0 into the FPGA Slaves address space (BSELselection).

LWFPGA Slaves Address Space

The Lightweight (LW) FPGA Slaves address space is located in the FPGA core and accessed from the HPS through the LWHPS2FPGA Bridge. The LWFPGA Slaves address space in the FPGA core is of unlimited size (soft logic in the FPGA performs address decoding). A portion of the Peripheral region provides a window of 2MB into the LWFPGA Slaves address space. The base address of the LWFPGA Slaves window is mapped to address 0x0 in the LWFPGA Slaves address space.

The HPS is not able to boot from the LWFPGA Slaves address space.

L3 Address Space

The L3 address space is 4 GB and applies to all L3 masters except the MPU. The L3 address space has more configuration options than the other address spaces.

Related Information

System Interconnect on page 7-1
For more information about configuring the L3 address space, refer to the System Interconnect chapter.

HPS Peripheral Region Address Map

Each peripheral slave interface has a dedicated address range in the peripheral region. The table below lists the base address and address range size for each slave.

Table 1-3: Peripheral Region Address Map

Slave Identifier	Description	Base Address	Size
STM	STM module	0xFC000000	48 MB
DAP	DAP module	0xFF000000	2 MB
LWFPGASLAVES	FPGA slaves accessed through lightweight HPS2FPGA bridge module	0xFF200000	2 MB
EMAC0	EMAC0 module	0xFF800000	8 KB
EMAC1	EMAC1 module	0xFF802000	8 KB
EMAC2	EMAC2 module	0xFF804000	8 KB
SDMMC	SD/MMC module	0xFF808000	4 KB
QSPIREGS	QSPI flash controller module registers	0xFF809000	4 KB
EMAC0RXECC	Receive ECC, Ethernet MAC0	0xFF8C0800	1 KB
EMAC0TXECC	Transmit ECC, Ethernet MAC0	0xFF8C0C00	1 KB
EMAC1RXECC	Receive ECC, Ethernet MAC1	0xFF8C1000	1 KB
EMAC1TXECC	Transmit ECC, Ethernet MAC1	0xFF8C1400	1 KB
EMAC2RXECC	Receive ECC, Ethernet MAC2	0xFF8C1800	1 KB
EMAC2TXECC	Transmit ECC, Ethernet MAC2	0xFF8C1C00	1 KB
NANDECC	NAND ECC	0xFF8C2000	1 KB
NANDREADECC	NAND read ECC	0xFF8C2400	1 KB
NANDWRITEECC	NAND write ECC	0xFF8C2800	1 KB
SDMMCECC	SD/MMC ECC	0xFF8C2C00	1 KB
OCRAMECC	On-chip RAM ECC	0xFF8C3000	1 KB
DMAECC	DMA ECC	0xFF8C8000	1 KB
QSPIECC	QSPI ECC	0xFF8C8400	1 KB
USB0ECC	USB 2.0 OTG 0 ECC	0xFF8C8800	1 KB
USB1ECC	USB 2.0 OTG 1 ECC	0xFF8C8C00	1 KB

Slave Identifier	Description	Base Address	Size
QSPIDATA	QSPI flash module data	0xFFA00000	1 MB
USB0	USB 2.0 OTG 0 controller module registers	0xFFB00000	256 KB
USB1	USB 2.0 OTG 1 controller module registers	0xFFB40000	256 KB
NANDREGS	NAND controller module registers	0xFFB80000	64 KB
NANDDATA	NAND controller module data	0xFFB90000	64 KB
UART0	UART0 module	0xFFC02000	256 B
UART1	UART1 module	0xFFC02100	256 B
I2C0	I^2C0 module	0xFFC02200	256 B
I2C1	I^2C1 module	0xFFC02300	256 B
I2C2	I^2C2 module (can be used with EMAC0)	0xFFC02400	256 B
I2C3	I^2C3 module (can be used with EMAC1)	0xFFC02500	256 B
I2C4	I^2C4 module (can be used with EMAC2)	0xFFC02600	256 B
SPTIMER0	SP Timer0 module	0xFFC02700	256 B
SPTIMER1	SP Timer1 module	0xFFC02800	256 B
GPIO0	GPIO0 module	0xFFC02900	256 B
GPIO1	GPIO1 module	0xFFC02A00	256 B
GPIO2	GPIO2 module	0xFFC02B00	256 B
HMCREGS	Hard memory controller registers	0xFFCFA000	4 KB
HMCAREGS	Hard memory controller adapter control registers	0xFFCFB000	4 KB
SECMGRDATA	Security manager module data	0xFFCFE000	1 KB
FPGAMGRDATA	FPGA manager module configuration data	0xFFCFE400	1 KB
OSC1TIMER0	OSC1 Timer0 module	0xFFD00000	256B
OSC1TIMER1	OSC1 Timer1 module	0xFFD00100	256B

Slave Identifier	Description	Base Address	Size
L4WD0	Watchdog0 module	0xFFD00200	256B
L4WD1	Watchdog1 module	0xFFD00300	256B
SECMGRREGS	Security manager module control and status registers	0xFFD02000	4 KB
FPGAMGRREGS	FPGA manager module control and status registers	0xFFD03000	4 KB
CLKMGR	Clock manager module	0xFFD04000	4 KB
RSTMGR	Reset manager module	0xFFD05000	4 KB
SYSMGR	System manager module	0xFFD06000	4 KB
IOMGR	I/O manager module	0xFFD07000	4 KB
FWL4PRIV	L4 privilege firewall registers	0xFFD11000	256 B
MPURADAPTER	MPU rate adapter registers	0xFFD11100	3.84 KB
DDRPRB	DDR probe registers	0xFFD12000	1 KB
SCHREGS	DDR scheduler control registers	0xFFD12400	128 B
FWL4PER	L4 peripheral firewall registers	0xFFD13000	256 B
FWL4SYS	L4 system firewall registers	0xFFD13100	256 B
FWOCRAM	On-chip RAM firewall registers	0xFFD13200	256 B
FWFPGA2SDRAM	DDR firewall registers for FPGA-to-SDRAM	0xFFD13300	256 B
FWDDRL3	DDR L3 firewall registers	0xFFD13400	256 B
FWHPS2FPGA	HPS-to-FPGA firewall registers	0xFFD13500	256 B
L4PRB	L4 bus probe registers	0xFFD14000	4 KB
MPUPRB	MPU probe and test registers	0xFFD15000	4 KB
L4QOS	L4 bus QoS	0xFFD16000	4 KB (estimated)
EMACTSF	EMAC transaction status filter registers	0xFFD1 7080	44 B

Slave Identifier	Description	Base Address	Size
DMANONSECURE	DMA non-secure module registers	0xFFDA0000	4 KB
DMASECURE	DMA secure module registers	0xFFDA1000	4 KB
SPI0	SPI module 0 slave	0xFFDA2000	4 KB
SPI1	SPI module 1 slave	0xFFDA3000	4 KB
SPI2	SPI module 0 master	0xFFDA4000	4 KB
SPI3	SPI module 1 master	0xFFDA5000	4 KB
OCRAM	On-chip RAM module	0xFFE00000	1 MB (256 KB used)
ROM	Boot ROM Module	0xFFFC0000	128 KB
MPU	MPU Module Registers	0xFFFFC000	8 KB
MPUL2	MPU L2 Cache Controller Module Registers	0xFFFFF000	4 KB

Document Revision History

Table 1-4: Document Revision History

Date	Version	Changes
October 2016	2016.10.28	Renamed MPU Subsystem to Cortex-A9 MPCore
May 2016	2016.05.27	Corrected the HPS-FPGA powering scheme.
May 2016	2016.05.03	Removed low-power double data rate 3 (LPDDR3) as a supported device.
November 2015	2015.11.02	Updated the link to the Memory Maps.
May 2015	2015.05.04	Corrected HPS-FPGA powering scheme.
December 2014	2014.12.15	Maintenance release
August 2014	2014.08.18	Initial release

Clock Manager 2

2016.10.28

a10_5v4

The hard processor system (HPS) clock generation is centralized in the clock manager. The clock manager is responsible for providing software-programmable clock control to configure all clocks generated in the HPS. Clocks are organized in clock groups. A clock group is a set of clock signals that originate from the same clock source. A phase-locked loop (PLL) clock group is a clock group where the clock source is a common PLL voltage-controlled oscillator (VCO).

Features of the Clock Manager

The *Clock Manager* offers the following features:

- Generates and manages clocks in the HPS
- Contains the following clock groups:
  - Main—contains clocks for the Cortex-A9 microprocessor unit (MPU) subsystem, level 3 (L3) interconnect, level 4 (L4) peripheral bus, and debug
  - Peripheral—contains clocks for PLL-driven peripherals
- Contains two identical 16-output PLL blocks:
  - PLL 0
  - PLL 1
- Generates clock gate controls for enabling and disabling most clocks
- Initializes and sequences clocks
- Allows software to program clock characteristics, such as the following items discussed later in this chapter:
  - Input clock source for the two PLLs
  - Multiplier range, divider range, and 16 post-scale counters for each PLL
  - VCO enable for each PLL
  - Bypass modes for each PLL
  - Gate of individual clocks in all PLL clock groups
  - Clear loss of lock status for each PLL
  - Boot mode for hardware-managed clocks
  - General-purpose I/O (GPIO) debounce clock divide

- Allows software to observe the status of all writable registers
- Supports interrupting the MPU subsystem on PLL-lock and loss-of-lock
- Supports clock gating at the signal level

The clock manager is **not** responsible for the following functional behaviors:

- Selection or management of the clocks for the FPGA-to-HPS and HPS-to-FPGA interfaces. The FPGA logic designer is responsible for selecting and managing these clocks.
- Software must not program the clock manager with illegal values. If it does, the behavior of the clock manager is undefined and could stop the operation of the HPS. The only guaranteed means for recovery from an illegal clock setting is a cold reset.
- When re-programming clock settings, there are no automatic glitch-free clock transitions. Software must follow a specific sequence to ensure glitch-free clock transitions. Refer to *Hardware-Managed and Software-Managed Clocks* section of this chapter.

Related Information

- **Hardware-Managed and Software-Managed Clocks** on page 2-11
- **Hardware-Managed and Software-Managed Clocks** on page 2-11

Clock Manager Block Diagram and System Integration

Figure 2-1: Clock Manager Block Diagram

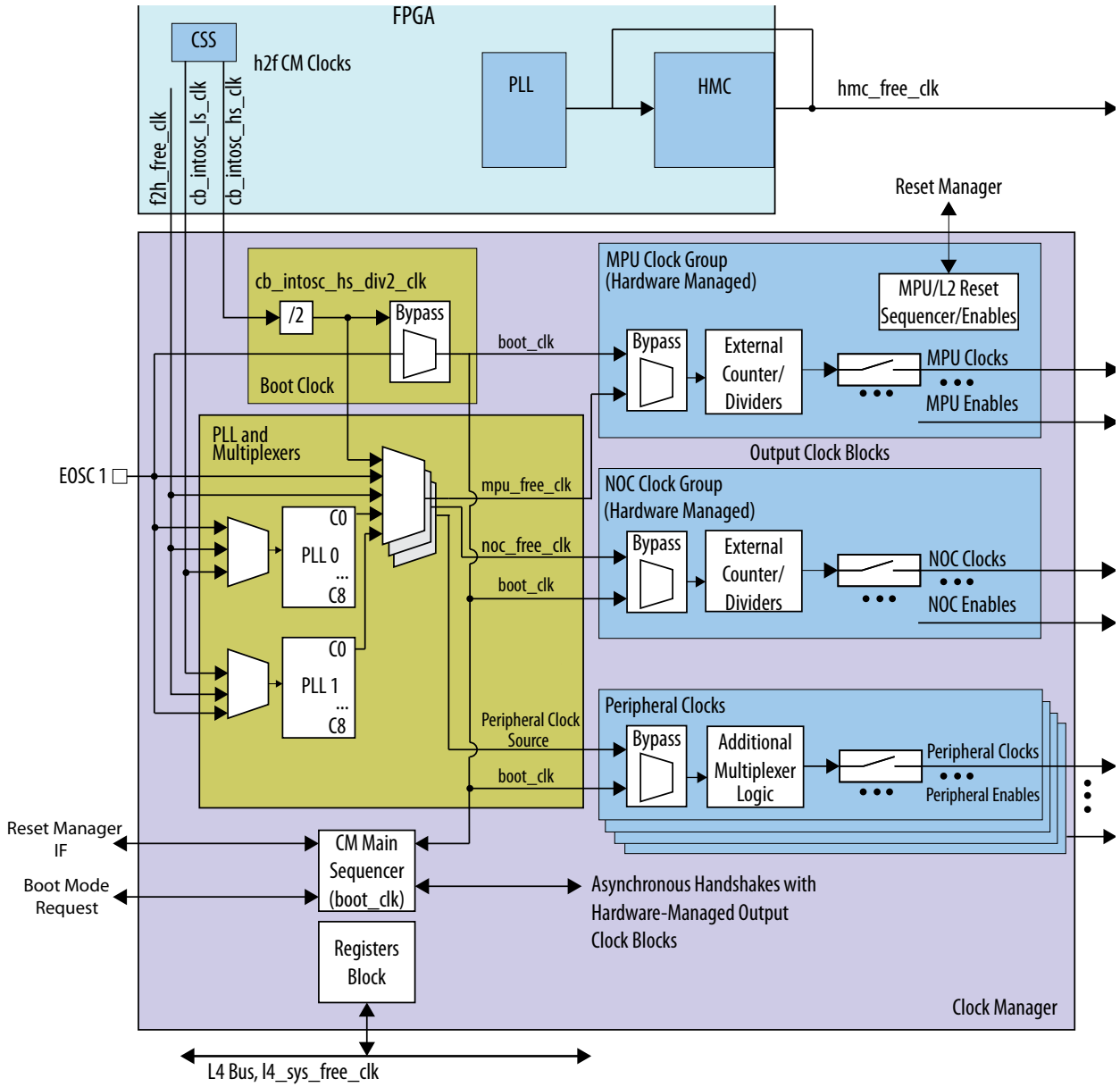


Table 2-1: Arria 10 Top Level Clocks

Clock Name	Source/Destination	Description
`mpu_free_clk`	Clock manager To MPU complex	Source clock from clock manager for both MPU clock groups.
`mpu_clk`	Internal to MPU complex	MPU main clock

Clock Name	Source/Destination	Description
mpu_l2ram_clk	Internal to MPU complex and HMC switch in NOC.	MPU L2 RAM clock and HMC switch in NOC. Fixed at ½ mpu_clk.
mpu_periph_clk	Internal to MPU complex	MPU peripherals clock for interrupts, timers, and watchdogs. Fixed at ¼ mpu_clk.
l3_main_free_clk	Clock manager to NOC/ Peripherals	Interconnect L3 main switch clock. Always free running.
l4_sys_free_clk	Clock manager to NOC/ Peripherals	Interconnect L4 system clock. Always free running.
l4_main_clk	Clock manager to NOC/ Peripherals	L4 Interconnect clock for fast peripherals including DMA, SPIM, SPIS and TCM.
l4_mp_clk	Clock manager to NOC/ Peripherals	Interconnect L4 peripheral clock.
l4_sp_clk	Clock manager to NOC/ Peripherals	Interconnect L4 slow peripheral clock.
cs_at_clk	Clock manager to CoreSight/ NOC	CoreSight Trace clock and debug time stamp clock.
cs_pdbg_clk	Clock manager to CoreSight	CoreSight bus clock
cs_trace_clk	Clock manager to CoreSight	CoreSight Trace I/O clock. This will be independent and defaults to a low frequency (25 MHz) for lower speed debuggers. Not required to be sync. to other clocks (but keep in clock group as hardware managed)
cs_timer_clk	Clock manager to CoreSight/ NOC	CoreSight timer clock. Same as cs_at_clk with different software enable to ensure MPU clock running. Not required to be sync. to other clocks (but still hardware managed).
fpga2soc_clk	FPGA fabric to NOC	FPGA to SoC interface clock from the FPGA.
soc2fpga_clk	FPGA fabric to NOC	SoC to FPGA interface clock from the FPGA.
lws2f_clk	FPGA fabric to NOC	LWHPS to FPGA interface clock from FPGA.
hmc_free_clk	From HMC to HMC switch in NOC	HMC interface clock from HMC (Hard Memory Controller) in FPGA.

Clock Name	Source/Destination	Description
f2h_sdram0_clk f2s_sdram1_clk f2h_sdram2_clk	FPGA fabric to HMC switch in NOC	FPGA fabric SDRAM write interfaces clocks.
f2h_pclkdbg	FPGA fabric to CoreSight bridge	CoreSight FPGA fabric APB debug port clock
usb[0,1]_ulpi_clk	I/O to USB	ULPI clock for PHY.

L4 Peripheral Clocks

The L4 peripheral clocks, denoted by l4_mp_clk, range up to 200 MHz.

Table 2-2: Clock List

Peripheral	Clock Name	Description
USB OTG 0/1(6)	hclk	AHB clock
	pmu_hclk	PMU AHB clock. pmu_hclk is the scan clock for the PMU's AHB domain. **Note:** Select it as a test clock.
	utmi_clk	Always used as the PHY domain clock during DFT Scan mode. **Note:** Select utmi_clk as a test clock even when the core is configured for a non-UTMI PHY.
Quad SPI Flash Controller(6)	pclk	APB clock
	hclk	AHB clock
NAND Flash Controller (Locally gated nand_mp_clk.)(6)	ACLK	AHB Data port clock
	mACLK	AXI Master port clock
	regACLK	AHB Register port clock
	ecc_clk	ECC circuitry clock
	clk_x	Bus Interface Clock
EMAC 0/1/2	aclk	Application clock for DMA AXI bus and CSR APB bus.
SD/MMC Controller	sdmmc_clk	All registers reside in the BIU clock domain.

For more information about the specific peripheral clocks, refer to their respective chapters.

Related Information

- **SD/MMC Controller** on page 14-1
- **NAND Flash Controller** on page 13-1
- **Quad SPI Flash Controller** on page 15-1
- **Ethernet Media Access Controller** on page 17-1
- **USB 2.0 OTG Controller** on page 18-1

Boot Clock

The Boot Clock (`boot_clk`) is used as the default clock for both cold or warm reset (Boot Mode), the Hardware Sequencer local clock and the external bypass clock reference.

The `boot_clk` is generated from the secure `cb_intosc_hs_div2_clk` or the unsecure external oscillator. `boot_clk` is only updated coming out of cold reset or a warm reset (boot mode request) and is not sampled at any other time.

The source of every output clock block comes from the following:

- Bypass source: `boot_clock` configured through fuses and security manager: registers at cold or warm reset
- Non-External Bypass: Each Clock may come from 1 of 5 sources:

Table 2-3: Non-External Bypass Sources of clocks

Source	Description
`OSC1`	Pin for external oscillator
`f2h_free_clk`	FPGA fabric PLL clock reference
`cb_intosc_hs_div2_clk`	FPGA CSS (Control Block) high speed internal oscillator divided by 2 (200 MHz maximum)
PLL0 Counter Output	Main PLL counter outputs 0 to 8
PLL1 Counter Output	Peripheral PLL counter outputs 0 to 8

Clock Output Blocks have the following features:

- Each clock output block contains an external bypass multiplexer
- Some clock output blocks contain additional dividers
- Clock gates controlled by either hardware or software are used to disable the clocks
- The MPU and NOC (includes debug clocks) blocks contain enable outputs to define clock frequency ratios to the MPU, NOC and CoreSight logic

The CSR Register logic uses an independent clock, `l4_sys_free_clk`, to allow the clock to be changed by software.

(6) Clock manager provides CSR bits for software enables to some peripherals. These enables are defaulted to enable. In boot mode, these enables are automatically active to ensure all clocks are active if RAM is cleared for security.

Updating PLL Settings without a System Reset

Updating the PLL settings wihtout a system reset will not reset the clock manager. All clocks will transition gracefully to their boot safe dividers. You can go in and out of boot mode and force the PLLs into bypass without doing a system reset. Software may reset the PLLs and reconfigure the VCO and bring it out of reset. By doing this you can safely update the PLL settings.

MPU Clock Scaling

You can dynamically slow down the MPU clock without touching the PLL by modifying the MPU external counter register (`MAINPLLGRP.MPUCLK.CNT`) to slow down the MPU frequency.

Related Information

Clock Manager Address Map and Register Definitions on page 2-18

Functional Description of the Clock Manager

Clock Manager Building Blocks

PLLs

The two PLLs in the clock manager generate the majority of clocks in the HPS. There is no phase control between the clocks generated by the two PLLs.

Each PLL has the following features:

- Phase detector and output lock signal generation
- Registers to set VCO frequency
  - Multiplier range is 1 to 4096
  - Divider range is 1 to 64
- 16 post-scale counters (C0-C8) with a range of 1 to 2048 to further subdivide the clock
- A PLL can be enabled to bypass all outputs to the input clock for glitch-free transitions

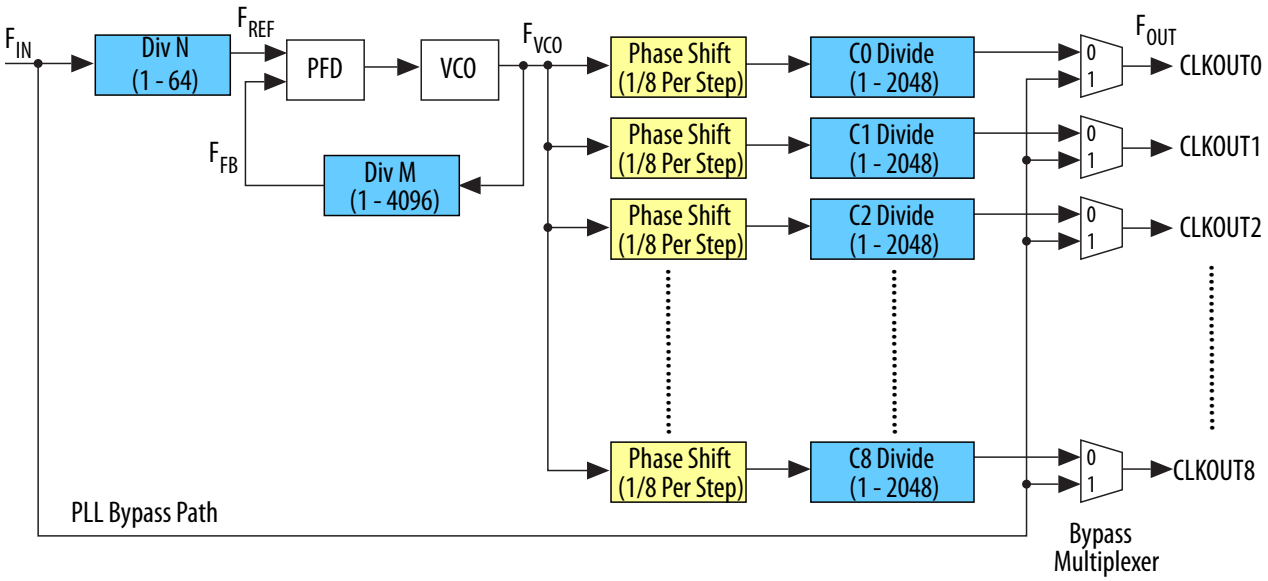
Related Information

- **PLL Integration** on page 2-9
- **Hardware Sequenced Clock Groups** on page 2-11
- **Software Sequenced Clocks** on page 2-14
- **Arria 10 Device Datasheet**

FREF, FVCO, and FOUT Equations

Figure 2-2: PLL Block Diagram

Values listed for M, N, and C are actually one greater than the values stored in the CSRs.



$$F_{REF} = F_{IN} / N$$
$$F_{VCO} = F_{REF} \times M = F_{IN} \times M/N$$
$$F_{OUT} = F_{VCO} /C_i = F_{REF} \times M/C_i = (F_{IN} \times M)/ (N \times C_i)$$

where:

- FVCO = VCO frequency
- FIN = input frequency
- FREF = reference frequency
- FOUT = output frequency
- M = numerator, part of the clock feedback path
- N = denominator, part of the input clock reference path
- Ci = post-scale counter, where *i* is 0-8

The Ci dividers are used to derive lower frequencies from the PLLs. The M and N dividers can be dynamically updated without losing the PLL lock if the VCO frequency changes less than 20%. If changes greater than 20% are needed, iteratively changing the frequency in increments of less than 20% allows a slow ramp of the VCO frequency without loss of clock.

To minimize jitter, use the following guidelines:

- The VCO should be as close to maximum as possible
- It is better to use the Ci dividers than the N divider. The N divider should be kept as small as possible.
- The M divider should be minimized, but should be greater than 32.

When making any changes that affect the VCO frequency, software must put the PLL into external bypass. After changing the VCO frequency, software must wait for the PLL lock, as indicated by the `intrs` register or enabled interrupt, before taking the PLL out of bypass.

As shown in the "Hardware Clock Groups" and "Peripheral Clocks" figures, every clock has five possible sources. When switching between clock sources:

- Put PLL into bypass mode
- Change the mux select
- Switch back out of bypass mode

Note: The new clock source must be active and locked before exiting bypass mode.

As shown in the "PLL Integration in Clock Manager" figure, each PLL has multiple input sources. If the input source is changed, the PLL loses VCO lock and all output clocks are not reliable. Before switching the PLL input source, software must select the `boot_clk` bypass for all PLL0 and PLL1 clocks. After the PLL output clocks are bypassed, software can change the PLL source and reinitialize the PLL.

Unused clock outputs should be set to a safe frequency such as 50 MHz to reduce power consumption and improve system stability.

Clock Gating

Clock gating enables and disables clock signals. Refer to the Peripheral PLL Group Enable Register (`en`) for more information on what clocks can be gated.

Related Information

Clock Manager Address Map and Register Definitions on page 2-18

PLL Integration

The two PLLs contain exactly the same set of output clocks. PLL0 is intended to be used for the MPU and the NOC clocks. PLL1 is intended to be used as the 250 MHz Ethernet clock reference.

Figure 2-3: PLL Integration in Clock Manager

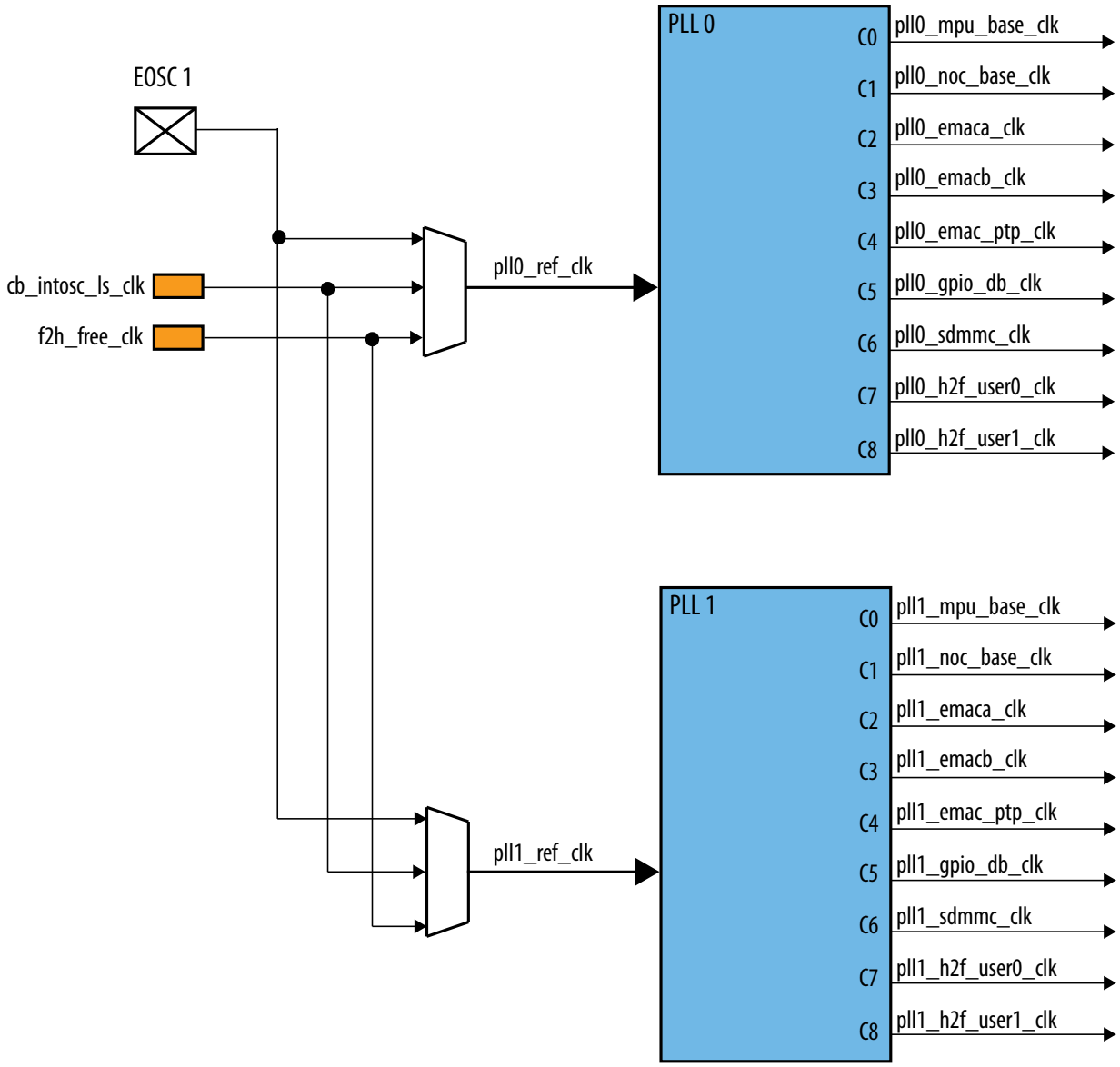


Table 2-4: PLL Output Clock Characteristics

Output Counter	Clock Name	Frequency	Boot Frequency
C0	`mpu_base_clk`	Up to varies	10 MHz to 200 MHz
C1	`noc_base_clk`	Up to C0/3	10 MHz to 200 MHz

(7) Frequency depends on device, refer to device datasheet for more information.

Output Counter	Clock Name	Frequency	Boot Frequency
C2	emaca_clk	50 to 250 MHz	10 MHz to 200 MHz
C3	emacb_clk	50 to 250 MHz	10 MHz to 200 MHz
C4	emac_ptp_clk	Up to 100 MHz	10 MHz to 200 MHz
C5	gpio_db_clk	Up to 200 MHz	10 MHz to 200 MHz
C6	sdmmc_clk	Up to 200 MHz	10 MHz to 200 MHz
C7	h2f_user0_clk	Up to 400 MHz	10 MHz to 200 MHz
C8	h2f_user1_clk	Up to 400 MHz	10 MHz to 200 MHz

Related Information

Arria 10 Device Datasheet

Hardware-Managed and Software-Managed Clocks

When changing values on clocks, the terms hardware-managed and software-managed define how clock transitions are implemented. When changing a software-managed clock, software is responsible for gating off the clock, waiting for a PLL lock if required, and gating the clock back on. Clocks that are hardware-managed are automatically transitioned by the hardware to ensure glitch-free operation.

- mpu_periph_clk
- mpu_l2_ram_clk
- mpu_clk

- l3_main_free_clk
- l4_sys_free_clk
- l4_sys_free_div4_clk
- l4_main_clk
- l4_mp_clk
- l4_sp_clk

- cs_timer_clk
- cs_at_clk
- cs_pdbg_clk
- cs_trace_clk

All other clocks in the HPS are software-managed clocks.

Hardware Sequenced Clock Groups

The hardware sequenced clock groups consists of the MPU clocks and the NOC clocks. The following diagram shows the external bypass muxes, hardware-managed external counters and dividers, and clock gates. For hardware-managed clocks, the group of clocks has only one software enable for the clock gate. As a result, the group of clocks are all enabled or disabled together. The slight exception is the NOC has two software enables, one for the l3/l4 clocks and one for the CoreSight clocks.

Figure 2-4: Hardware Clock Groups

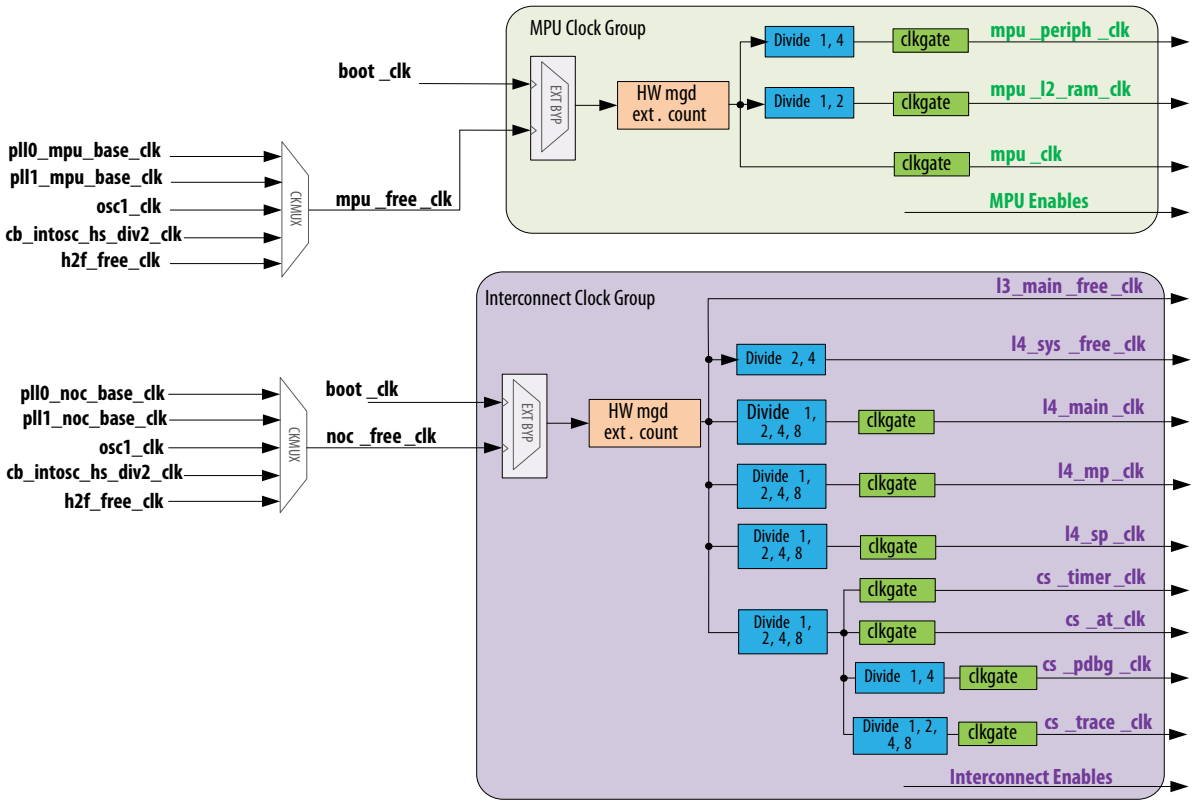


Table 2-5: The Hardware Sequenced Clocks Feature Summary

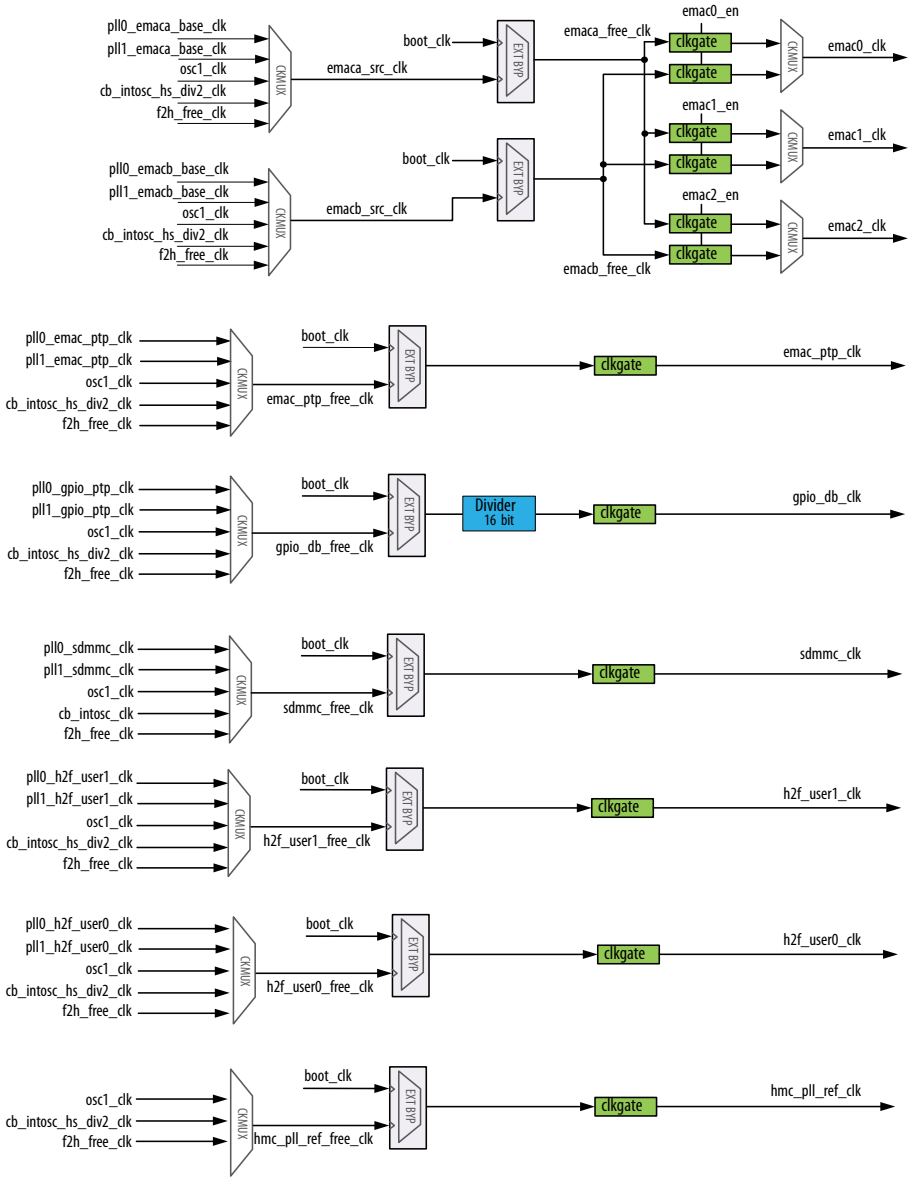
Clock Output Group	System Clock Name	Frequency	Boot Frequency	Uses
MPU	mpu_clk	PLL C0	boot_clk	MPU subsystem, including CPU0 and CPU1
	mpu_I2_ram_clk	mpu_clk/2	boot_clk	MPU level 2 (L2) RAM
	mpu_periph_clk	mpu_clk/4	boot_clk	MPU snoop control unit (SCU) peripherals, such as the general interrupt controller (GIC)

Clock Output Group	System Clock Name	Frequency	Boot Frequency	Uses
NOC	l3_main_free_clk	PLL C1	boot_clk	L3 interconnect
	l4_sys_free_clk	l3_main_free_clk/4	boot_clk/2	L4 interconnect
	l4_sys_free_div4_clk	l4_sys_free_clk/4	l4_sys_free_clk/4	L4 interconnect timer reference
	l4_main_clk	l3_main_free_clk/{1,2,4,8}	boot_clk	L4 main bus
	l4_mp_clk	l3_main_free_clk/{1,2,4,8}	boot_clk	L4 MP bus
	l4_sp_clk	l3_main_free_clk/{1,4,8}	boot_clk/2	L4 SP bus
	cs_timer_clk	l3_main_free_clk/{1,2,4,8}	boot_clk	Trace timestamp generator
	cs_at_clk	l3_main_free_clk/{1,2,4,8}	boot_clk	CoreSight debug trace bus
	cs_pdbg_clk	cs_at_clk/{1,4}	cs_at_clk/2	Debug Access Port (DAP) and debug peripheral bus
	cs_trace_clk	cs_at_clk/{1,2,8}	cs_at_clk/4	Coresight debug Trace Port Interface Unit (TPIU)
Main FPGA Reference	h2f_user0_clk	h2f_user0_free_clk	boot_clk	FPGA
Main HMC PLL Reference	hmc_pll_ref_clk	hmc_pll_ref_free_clk	boot_clk	HMC FPGA IO PLL

Software Sequenced Clocks

The software sequenced clock groups include additional clocks for peripherals not covered by the NOC clocks. The main purpose is to have a second PLL for the Ethernet 250 MHz clock reference. The following diagram shows the external bypass muxes, hardware-managed external counters and dividers, and clock gates.

Figure 2-5: Peripheral Clocks



There are 3 EMAC cores that have a very strict requirement of either a 250 MHz or 50 MHz clock reference. If the PLL0 frequency is a multiple of 250 MHz (for example 1.5 GHz), driving the EMAC clocks from PLL0 provides PLL1 with more flexibility in VCO clock frequency. In addition, to minimize the PLL clock outputs required, `emac_clka` can be 250 MHz and `emac_clkb` can be 50 MHz, allowing each EMAC core to be software configured to select 250 MHz or 50 MHz.

Table 2-6: Software Sequenced Clocks Feature Summary

System Clock Name	Frequency	Boot Frequency	Descriptions
emac{0,1,2}_clk	PLL C2 or PLL C3	boot_clk	Clock for EMAC. Fixed at 250 MHz or 250 MHz emac_clk and 50 MHz emacb_clk
emac_ptp_clk	PLL C4	boot_clk	Clock for EMAC PTP timestamp clock
gpio_db_clk	125 Hz to PLL C5	boot_clk	Clock for GPIO debounce clock
sdmmc_clk	PLL C6	boot_clk	Clock for SDMMC
h2f_user0_clk	PLL C7	boot_clk	Clock reference for FPGA
h2f_user1_clk	PLL C8	boot_clk	Clock reference for FPGA

Resets

Power-On-Reset (POR) Reset

The security manager handles clocking during POR. Depending on the status of one of the security fuses, the device is initially clocked using either the secure cb_intosc_hs_div2_clk or the unsecure external oscillator.

Cold Reset

The reset manager brings the clock manager out of cold reset first in order to provide clocks to the rest of the blocks. After POR is de-asserted, clock manager enables boot_clk to the rest of the system before the module resets are de-asserted.

Warm Reset

During a Warm Rest, the clock manager module is not reset. The following steps are taken during a warm reset:

1. Reset manager puts all modules affected by Warm Reset into reset. Reset manager issues a Boot Mode request to clock manager.
2. Based on the status of the hps_clk_f fuse during POR, security manager indicates if the boot clock should be secure.
   a. If secure clocks are enabled, boot_clk transitions gracefully to cb_intosc_hs_div2_clk.
   b. If secure clocks are not enabled, boot_clk transitions gracefully to EOSC1.

   Note: The security fuse is only sampled during cold reset and warm reset. The security fuse HPS CLK allows the user to enable secure clocks. If clearing RAM on a Cold or Warm reset, the user should enable secure clocks (cb_intosc_hs_clk divide by 2).
3. The *Clock Manager* gracefully transitions Hardware-Managed and Software Managed clocks into Boot Mode as follows:
   a. Disable all output clocks including Hardware and Software-Managed clocks.
   b. Wait for all clocks to be disabled, and do the following two things:

   a. Bypass all external Hardware and Software-Managed clocks.
   b. Update Hardware-Managed external counters/dividers to Boot Mode settings.
c. Wait for all bypasses to switch, and then synchronously reset the CSR registers.
d. Enable all clocks.

4. After Hardware Managed Clocks have transitioned, the *Clock Manager* acknowledges the *Reset Manager*.
5. *Reset Manager* continues with Warm Reset de-assertion sequence.

Related Information

SoC Security on page 6-1

Security

The clock manager creates a boot clock as the clock reference for boot mode and external bypass. The clock configuration is based on security features in the security manager.

Security Input Clocks

The following table defines the boot clock sources.

Table 2-7: Security Input Clocks

Clock Name	Max	Min	Source/Dest	Description
`EOSC1`	50	10	Pin	External Oscillator Clock Reference. Non-secure clock reference.
`cb_intosc_hs_clk`	400	120	FPGA Control Block	Control Block high speed internal ring oscillator. This clock has wide variation across process/temperature. This clock is used as the secure reference for Boot Mode. Because the range/jitter of the clock is low quality, the clock is divided by 2.

Clock Name	Max	Min	Source/Dest	Description
cb_intosc_ls_clk	100	30	FPGA Control Block	Control Block low speed internal ring oscillator. This clock has wide variation across process/ temperature. This clock is used by *Security Manager* as the Power on Reset Domain secure clock. This clock is also provided as an input to the *Clock Manager* PLLs. However, because of the low quality of the clock, it is not recommended to use this clock as the PLL reference.

Boot Clock

The status of the hps_clk_f security fuse in the *Security Manager* determines if boot_clk should be secure:

- If set, then boot_clk is cb_intosc_hs_clk divided by 2.
- If clear, then boot_clk is EOSC1.

The above setting is reported by the security manager before the chip is brought out of cold reset. Also the security status may be updated by security option registers in the security manager.

Related Information

SoC Security on page 6-1

Interrupts

The clock manager provides one interrupt output, which is enabled through the interrupt enable register (intren). The interrupt can be programmed to trigger twhen either the PLL achieves or loses its lock.

Clock Manager Address Map and Register Definitions

For complete HPS address map and register definitions, refer to the **Arria 10 HPS Address Map and Register Definitions**.

clkmgr_clkmgrp Address Map

Module Instance	Base Address	End Address
i_clk_mgr_clkmgr	0xFFD04000	0xFFD0403F

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
ctrl on page 2-21	0x0	32	RW	0x3	Control Register
intr on page 2-23	0x4	32	RW	0x0	Interrupt Status Register
intrs on page 2-25	0x8	32	RW	0x0	Interrupt Status Register Set
intrr on page 2-27	0xC	32	RW	0x0	Interrupt Status Register Reset
intren on page 2-29	0x10	32	RW	0x0	Interrupt Enable Register
intrens on page 2-31	0x14	32	RW	0x0	Interrupt Enable Register Set
intrenr on page 2-33	0x18	32	RW	0x0	Interrupt Enable Register Reset
stat on page 2-35	0x1C	32	RO	0x10000	Status Register
testioctrl on page 2-37	0x20	32	RW	0x100808	Test IO Control Register

clkmgr_clkmgrp Summary

Base Address: 0xFFD04000

Register Address Offset	Bit Fields
i_clk_mgr_clkmgr	

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ctrl 0x0	Reserved						swctrlbtclksel RW 0x0	swctrlbtclken RW 0x0	Reserved							bootmode RW 0x1
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
intr 0x4	Reserved				perllfbslip RW 0x0	mainpllfbslip RW 0x0	perllrfslip RW 0x0	mainpllrfslip RW 0x0	Reserved				perllllost RW 0x0	mainplllost RW 0x0	perllachieved RW 0x0	mainpllachieved RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
intrs 0x8	Reserved				perllfbslip RW 0x0	mainpllfbslip RW 0x0	perllrfslip RW 0x0	mainpllrfslip RW 0x0	Reserved				perllllost RW 0x0	mainplllost RW 0x0	perllachieved RW 0x0	mainpllachieved RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
intrr 0xC	Reserved				perllfbslip RW 0x0	mainpllfbslip RW 0x0	perllrfslip RW 0x0	mainpllrfslip RW 0x0	Reserved				perllllost RW 0x0	mainplllost RW 0x0	perllachieved RW 0x0	mainpllachieved RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
intren 0x10	Reserved				perllfbslip RW 0x0	mainpllfbslip RW 0x0	perllrfslip RW 0x0	mainpllrfslip RW 0x0	Reserved				perllllost RW 0x0	mainplllost RW 0x0	perllachieved RW 0x0	mainpllachieved RW 0x0

Register Address Offset	Bit Fields															
intrens 0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				perpllfbslip RW 0x0	mainpllfbslip RW 0x0	perpllrfslip RW 0x0	mainpllrfslip RW 0x0	Reserved				perplllost RW 0x0	mainplllost RW 0x0	perpllachieved RW 0x0	mainpllachieved RW 0x0
intrenr 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				perpllfbslip RW 0x0	mainpllfbslip RW 0x0	perpllrfslip RW 0x0	mainpllrfslip RW 0x0	Reserved				perplllost RW 0x0	mainplllost RW 0x0	perpllachieved RW 0x0	mainpllachieved RW 0x0
stat 0x1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved														bootclksrc RO 0x0	bootmode RO 0x1
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						perplllocked RO 0x0	mainplllocked RO 0x0	Reserved							busy RO 0x0
testioctrl 0x20	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved											debugclksel RW 0x10				
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				periclksel RW 0x8				Reserved				mainclksel RW 0x8			

ctrl

Contains fields that control the entire Clock Manager.

Bit	Name	Description	Access	Reset
9	swctrlbtclksel	This bit is only used if swctrlbtclken is set. If 1, boot_clk source will be from cb_intosc_hs_clk divided by 2. If 0, boot_clk source will be from the external oscillator (EOSC1). This bit is cleared on a cold reset. Warm reset has no affect on this bit.	RW	0x0
8	swctrlbtclken	If set, then Software will take control of the boot_clk mux select. If set, then swctrlbtclksel will determine the mux setting. If not set, the security features will determine the fuse settings. This bit is cleared on a cold reset. Warm reset has no affect on this bit.	RW	0x0

Bit	Name	Description	Access	Reset
0	bootmode	When set the Clock Manager is in Boot Mode. In Boot Mode Clock Manager register settings defining clock behavior are ignored and clocks are set to their Boot Mode settings. All clocks will be bypassed and external HW managed counters and dividers will be set to divide by 1. This bit should only be cleared when clocks have been correctly configured. This field is set on a cold reset and optionally on a warm reset. SW may set this bit to force the clocks into Boot Mode. SW exits Boot Mode by clearing this bit.	RW	0x1

intr

Contains interrupt fields for Clock Manager

Bit	Name	Description	Access	Reset
11	perpllfbslip	If 1, the Peripheral PLL feedback cycle has slipped (CLKOUT frequency too low). This does not mean the PLL has lost lock, but the quality of the clock has degraded.	RW	0x0
10	mainpllfbslip	If 1, the Main PLL feedback cycle has slipped (CLKOUT frequency too low). This does not mean the PLL has lost lock, but the quality of the clock has degraded.	RW	0x0
9	perpllrfslip	If 1, the Peripheral PLL reference cycle has slipped (CLKOUT frequency too high). This does not mean the PLL has lost lock, but the quality of the clock has degraded.	RW	0x0
8	mainpllrfslip	If 1, the Main PLL reference cycle has slipped (CLKOUT frequency too high). This does not mean the PLL has lost lock, but the quality of the clock has degraded.	RW	0x0

Bit	Name	Description	Access	Reset
3	perplllost	If 1, the Peripheral PLL has lost lock at least once since this bit was cleared. If 0, the Peripheral PLL has not lost lock since this bit was cleared.	RW	0x0
2	mainplllost	If 1, the Main PLL has lost lock at least once since this bit was cleared. If 0, the Main PLL has not lost lock since this bit was cleared.	RW	0x0
1	perpllachieved	If 1, the Peripheral PLL has achieved lock at least once since this bit was cleared. If 0, the Peripheral PLL has not achieved lock since this bit was cleared.	RW	0x0
0	mainpllachieved	If 1, the Main PLL has achieved lock at least once since this bit was cleared. If 0, the Main PLL has not achieved lock since this bit was cleared.	RW	0x0

intrs

Contains fields that indicate the PLL lock status.

Bit	Name	Description	Access	Reset
11	perpllfbslip	If 1, the Peripheral PLL feedback cycle has slipped (CLKOUT frequency too low). This does not mean the PLL has lost lock, but the quality of the clock has degraded.	RW	0x0
10	mainpllfbslip	If 1, the Main PLL feedback cycle has slipped (CLKOUT frequency too low). This does not mean the PLL has lost lock, but the quality of the clock has degraded.	RW	0x0
9	perpllrfslip	If 1, the Peripheral PLL reference cycle has slipped (CLKOUT frequency too high). This does not mean the PLL has lost lock, but the quality of the clock has degraded.	RW	0x0
8	mainpllrfslip	If 1, the Main PLL reference cycle has slipped (CLKOUT frequency too high). This does not mean the PLL has lost lock, but the quality of the clock has degraded.	RW	0x0

Bit	Name	Description	Access	Reset
3	perplllost	If 1, the Peripheral PLL has lost lock at least once since this bit was cleared. If 0, the Peripheral PLL has not lost lock since this bit was cleared.	RW	0x0
2	mainplllost	If 1, the Main PLL has lost lock at least once since this bit was cleared. If 0, the Main PLL has not lost lock since this bit was cleared.	RW	0x0
1	perpllachieved	If 1, the Peripheral PLL has achieved lock at least once since this bit was cleared. If 0, the Peripheral PLL has not achieved lock since this bit was cleared.	RW	0x0
0	mainpllachieved	If 1, the Main PLL has achieved lock at least once since this bit was cleared. If 0, the Main PLL has not achieved lock since this bit was cleared.	RW	0x0

intrr

Contains fields that indicate the PLL lock status.

Module Instance	Base Address	Register Address
i_clk_mgr_clkmgr	0xFFD04000	0xFFD0400C

Offset: 0xC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				perpllfbslip RW 0x0	mainpllfbslip RW 0x0	perpllrfslip RW 0x0	mainpllrfslip RW 0x0	Reserved				perpllost RW 0x0	mainplllost RW 0x0	perpllachieved RW 0x0	mainpllachieved RW 0x0

intrr Fields

Bit	Name	Description	Access	Reset
11	perpllfbslip	If 1, the Peripheral PLL feedback cycle has slipped (CLKOUT frequency too low). This does not mean the PLL has lost lock, but the quality of the clock has degraded.	RW	0x0
10	mainpllfbslip	If 1, the Main PLL feedback cycle has slipped (CLKOUT frequency too low). This does not mean the PLL has lost lock, but the quality of the clock has degraded.	RW	0x0
9	perpllrfslip	If 1, the Peripheral PLL reference cycle has slipped (CLKOUT frequency too high). This does not mean the PLL has lost lock, but the quality of the clock has degraded.	RW	0x0
8	mainpllrfslip	If 1, the Main PLL reference cycle has slipped (CLKOUT frequency too high). This does not mean the PLL has lost lock, but the quality of the clock has degraded.	RW	0x0

Bit	Name	Description	Access	Reset
3	perplllost	If 1, the Peripheral PLL has lost lock at least once since this bit was cleared. If 0, the Peripheral PLL has not lost lock since this bit was cleared.	RW	0x0
2	mainplllost	If 1, the Main PLL has lost lock at least once since this bit was cleared. If 0, the Main PLL has not lost lock since this bit was cleared.	RW	0x0
1	perpllachieved	If 1, the Peripheral PLL has achieved lock at least once since this bit was cleared. If 0, the Peripheral PLL has not achieved lock since this bit was cleared.	RW	0x0
0	mainpllachieved	If 1, the Main PLL has achieved lock at least once since this bit was cleared. If 0, the Main PLL has not achieved lock since this bit was cleared.	RW	0x0

intren

Contain fields that enable the interrupt

Module Instance	Base Address	Register Address
i_clk_mgr_clkmgr	0xFFD04000	0xFFD04010

Offset: 0x10

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				perpllfbslip RW 0x0	mainpllfbslip RW 0x0	perpllrfslip RW 0x0	mainpllrfslip RW 0x0	Reserved				perpllost RW 0x0	mainplllost RW 0x0	perpllachieved RW 0x0	mainpllachieved RW 0x0

intren Fields

Bit	Name	Description	Access	Reset
11	perpllfbslip	When set to 1,the Peripheral PLL feedback cycle slipped bit is ORed into the Clock Manager interrupt output. When set to 0, the Peripheral PLL feedback cylce slipped bit is not ORed into the Clock Manager interrupt output.	RW	0x0
10	mainpllfbslip	When set to 1,the Main PLL feedback cycle slipped bit is ORed into the Clock Manager interrupt output. When set to 0, the Main PLL feedback cylce slipped bit is not ORed into the Clock Manager interrupt output.	RW	0x0
9	perpllrfslip	When set to 1,the Peripheral PLL reference cycle slipped bit is ORed into the Clock Manager interrupt output. When set to 0, the Peripheral PLL reference cylce slipped bit is not ORed into the Clock Manager interrupt output.	RW	0x0
8	mainpllrfslip	When set to 1,the Main PLL reference cycle slipped bit is ORed into the Clock Manager interrupt output. When set to 0, the Main PLL reference cylce slipped bit is not ORed into the Clock Manager interrupt output.	RW	0x0

Bit	Name	Description	Access	Reset
3	perplllost	When set to 1, the Peripheral PLL lost lock bit is ORed into the Clock Manager interrupt output. When set to 0 the Peripheral PLL lost lock bit is not ORed into the Clock Manager interrupt output.	RW	0x0
2	mainplllost	When set to 1, the Main PLL lost lock bit is ORed into the Clock Manager interrupt output. When set to 0 the Main PLL lost lock bit is not ORed into the Clock Manager interrupt output.	RW	0x0
1	perpllachieved	When set to 1, the Peripheral PLL achieved lock bit is ORed into the Clock Manager interrupt output. When set to 0 the Peripheral PLL achieved lock bit is not ORed into the Clock Manager interrupt output.	RW	0x0
0	mainpllachieved	When set to 1, the Main PLL achieved lock bit is ORed into the Clock Manager interrupt output. When set to 0 the Main PLL achieved lock bit is not ORed into the Clock Manager interrupt output.	RW	0x0

intrens

Contain fields that enable the interrupt

Module Instance	Base Address	Register Address
i_clk_mgr_clkmgr	0xFFD04000	0xFFD04014

Offset: 0x14

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				perpllfbslip RW 0x0	mainpllfbslip RW 0x0	perpllrfslip RW 0x0	mainpllrfslip RW 0x0	Reserved				perpllost RW 0x0	mainpllost RW 0x0	perpllachieved RW 0x0	mainpllachieved RW 0x0

intrens Fields

Bit	Name	Description	Access	Reset
11	perpllfbslip	When set to 1,the Peripheral PLL feedback cycle slipped bit is ORed into the Clock Manager interrupt output. When set to 0, the Peripheral PLL feedback cylce slipped bit is Ored into the Clock Manager interrupt output.	RW	0x0
10	mainpllfbslip	When set to 1,the Main PLL feedback cycle slipped bit is ORed into the Clock Manager interrupt output. When set to 0, the Main PLL feedback cylce slipped bit is Ored into the Clock Manager interrupt output.	RW	0x0
9	perpllrfslip	When set to 1,the Peripheral PLL reference cycle slipped bit is ORed into the Clock Manager interrupt output. When set to 0, the Peripheral PLL reference cylce slipped bit is Ored into the Clock Manager interrupt output.	RW	0x0
8	mainpllrfslip	When set to 1,the Main PLL reference cycle slipped bit is ORed into the Clock Manager interrupt output. When set to 0, the Main PLL reference cylce slipped bit is Ored into the Clock Manager interrupt output.	RW	0x0

Bit	Name	Description	Access	Reset
3	perplllost	When set to 1, the Peripheral PLL lost lock bit is ORed into the Clock Manager interrupt output. When set to 0 the Peripheral PLL lost lock bit is not ORed into the Clock Manager interrupt output.	RW	0x0
2	mainplllost	When set to 1, the Main PLL lost lock bit is ORed into the Clock Manager interrupt output. When set to 0 the Main PLL lost lock bit is not ORed into the Clock Manager interrupt output.	RW	0x0
1	perpllachieved	When set to 1, the Peripheral PLL achieved lock bit is ORed into the Clock Manager interrupt output. When set to 0 the Peripheral PLL achieved lock bit is not ORed into the Clock Manager interrupt output.	RW	0x0
0	mainpllachieved	When set to 1, the Main PLL achieved lock bit is ORed into the Clock Manager interrupt output. When set to 0 the Main PLL achieved lock bit is not ORed into the Clock Manager interrupt output.	RW	0x0

intrenr

Contain fields that enable the interrupt

Module Instance	Base Address	Register Address
i_clk_mgr_clkmgr	0xFFD04000	0xFFD04018

Offset: 0x18

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				perpllfbslip RW 0x0	mainpllfbslip RW 0x0	perpllrfslip RW 0x0	mainpllrfslip RW 0x0	Reserved				perpllost RW 0x0	mainpllost RW 0x0	perpllachieved RW 0x0	mainpllachieved RW 0x0

intrenr Fields

Bit	Name	Description	Access	Reset
11	perpllfbslip	When set to 1,the Peripheral PLL feedback cycle slipped bit is ORed into the Clock Manager interrupt output. When set to 0, the Peripheral PLL feedback cylce slipped bit is Ored into the Clock Manager interrupt output.	RW	0x0
10	mainpllfbslip	When set to 1,the Main PLL feedback cycle slipped bit is ORed into the Clock Manager interrupt output. When set to 0, the Main PLL feedback cylce slipped bit is Ored into the Clock Manager interrupt output.	RW	0x0
9	perpllrfslip	When set to 1,the Peripheral PLL reference cycle slipped bit is ORed into the Clock Manager interrupt output. When set to 0, the Peripheral PLL reference cylce slipped bit is Ored into the Clock Manager interrupt output.	RW	0x0
8	mainpllrfslip	When set to 1,the Main PLL reference cycle slipped bit is ORed into the Clock Manager interrupt output. When set to 0, the Main PLL reference cylce slipped bit is Ored into the Clock Manager interrupt output.	RW	0x0

Bit	Name	Description	Access	Reset
3	perplllost	When set to 1, the Peripheral PLL lost lock bit is ORed into the Clock Manager interrupt output. When set to 0 the Peripheral PLL lost lock bit is not ORed into the Clock Manager interrupt output.	RW	0x0
2	mainplllost	When set to 1, the Main PLL lost lock bit is ORed into the Clock Manager interrupt output. When set to 0 the Main PLL lost lock bit is not ORed into the Clock Manager interrupt output.	RW	0x0
1	perpllachieved	When set to 1, the Peripheral PLL achieved lock bit is ORed into the Clock Manager interrupt output. When set to 0 the Peripheral PLL achieved lock bit is not ORed into the Clock Manager interrupt output.	RW	0x0
0	mainpllachieved	When set to 1, the Main PLL achieved lock bit is ORed into the Clock Manager interrupt output. When set to 0 the Main PLL achieved lock bit is not ORed into the Clock Manager interrupt output.	RW	0x0

stat

Provides status for Clock Manager including PLL lock and HW Managed Clock State Machine busy.

Module Instance	Base Address	Register Address
i_clk_mgr_clkmgr	0xFFD04000	0xFFD0401C

Offset: 0x1C

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved														bootclksrc RO 0x0	bootmode RO 0x1
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						perpllocked RO 0x0	mainplllocked RO 0x0	Reserved							busy RO 0x0

stat Fields

Bit	Name	Description	Access	Reset
17	bootclksrc	If 1, the source of boot_clk is cb_intosc_hs_div2_clk. . If 0, the boot_clk source is the external oscillator (EOSC1). This is a read only status.	RO	0x0
16	bootmode	If 1, the clocks are currently in Boot Mode. If 0, the clocks are not in Boot Mode. This is a read only status. For SW to exit Boot Mode, SW must clear the RW bit CTRL.BOOTMODE.	RO	0x1
9	perplllocked	If 1, the Peripheral PLL is currently locked. If 0, the Peripheral PLL is currently not locked.	RO	0x0
8	mainplllocked	If 1, the Main PLL is currently locked. If 0, the Main PLL is currently not locked.	RO	0x0

Bit	Name	Description	Access	Reset
0	busy	This read only bit indicates that the Hardware Managed clock's state machine is active. If the state machine is active, then the clocks are in transition. Software should poll this bit after changing the source of internal clocks when changing the state of CTRL.BOOTMODE, MAINPLLGRP.BYPASS.MPU or MAINPLLGRP.BYPASS.NOC register bits. Immediately following writes to any of these registers, SW should wait 0.5 usecs and then poll this BUSY bit until it is IDLE before proceeding with any other register writes in the Clock Manager. The reset value of this bit is applied on a cold reset; warm reset has no affect on this bit. **Value** **Description** 0 IDLE 1 BUSY	RO	0x0

testioctrl

Contains fields setting the IO output select for Test Clock and Debug outputs. The dedicated IO outputs includes two outputs for the Main PLL clock outputs (PLL_CLK0 and PLL_CLK1), two outputs for the Peripheral PLL clock outputs (PLL_CLK2 and PLL_CLK3), and one output for miscelaneous debug for the Main and Peripheral PLL (PLL_CLK4).

The Test Clock and Debug outputs will only propagate to the dedicated IO based on the IO pinmux configuration. If Test Clocks are selected in the pinmux, then the selects in this register determine which PLL clocks and PLL debug signals will propagate to the IOs.

Module Instance	Base Address	Register Address
i_clk_mgr_clkmgr	0xFFD04000	0xFFD04020

Offset: 0x20

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved											debugclksel RW 0x10				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				periclksel RW 0x8				Reserved				mainclksel RW 0x8			

testioctrl Fields

Bit	Name	Description	Access	Reset
20:16	debugclksel	Selects the source of PLL_CLK4 for miscellaneous PLL signals. Bit[3] (p below) determines if from the debug output is from the Main or Peripheral PLL. If 0, the the output is from the Main PLL and if 1, the output is from the Peripheral PLL. The following table determines the PLL debug output select: sel PLL_CLK4 0p000 OUTRESETACK0 0p001 OUTRESETACK3 0p010 OUTRESETACK7 0p011 PLLRESET 0p100 OUTRESETACK15 0p101 FBSLIP 0p110 RFSLIP 0p111 LOCK 1xxxx VSS	RW	0x10

Bit	Name	Description	Access	Reset
11:8	periclksel	Selects the source of PLL_CLK2 and PLL_CLK3 dedicated IO outputs if selected. All of the CLKOUT# counter outputs are from the Peripheral PLL. The following table determines the PLL counter output select: sel PLL_CLK2 PLL_CLK3 0000 CLKOUT0 CLKOUT8 0001 CLKOUT1 CLKOUT9 0010 CLKOUT2 CLKOUT10 0011 CLKOUT3 CLKOUT11 0100 CLKOUT4 CLKOUT13 0101 CLKOUT5 CLKOUT14 0110 CLKOUT6 CLKOUT15 0111 CLKOUT7 CLKOUT16 1xxx VSS VSS	RW	0x8
3:0	mainclksel	Selects the source of PLL_CLK0 and PLL_CLK1 dedicated IO outputs if selected. All of the CLKOUT# counter outputs are from the Main PLL. The following table determines the PLL counter output select: sel PLL_CLK0 PLL_CLK1 0000 CLKOUT0 CLKOUT8 0001 CLKOUT1 CLKOUT9 0010 CLKOUT2 CLKOUT10 0011 CLKOUT3 CLKOUT11 0100 CLKOUT4 CLKOUT13 0101 CLKOUT5 CLKOUT14 0110 CLKOUT6 CLKOUT15 0111 CLKOUT7 CLKOUT16 1xxx VSS VSS	RW	0x8

clkmgr_mainpllgrp Address Map

Module Instance	Base Address	End Address
i_clk_mgr_mainpllgrp	0xFFD04040	0xFFD040BF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
vco0 on page 2-45	0x0	32	RW	0x10043	Main PLL VCO Control Register 0
vco1 on page 2-47	0x4	32	RW	0x10001	Main PLL VCO Control Register 1
en on page 2-48	0x8	32	RW	0xFF	Enable Register
ens on page 2-49	0xC	32	RW	0xFF	Enable Set Register
enr on page 2-50	0x10	32	RW	0xFF	Enable Reset Register
bypass on page 2-52	0x14	32	RW	0x3F	Bypass Register
bypasss on page 2-53	0x18	32	RW	0x3F	Bypass Set Register
bypassr on page 2-54	0x1C	32	RW	0x3F	Bypass Reset Register
mpuclk on page 2-55	0x20	32	RW	0x0	Main PLL Control Register for MPU Clock Group.
nocclk on page 2-56	0x24	32	RW	0x0	Main PLL Control Register for NOC Clock Group.
cntr2clk on page 2-57	0x28	32	RW	0x0	Main PLL Control Register for Counter 2 Clock
cntr3clk on page 2-58	0x2C	32	RW	0x0	Main PLL Control Register for Counter 3 Clock
cntr4clk on page 2-59	0x30	32	RW	0x0	Main PLL Control Register for Counter 4 Clock
cntr5clk on page 2-59	0x34	32	RW	0x0	Main PLL Control Register for Counter 5 Clock

Register	Offset	Width	Access	Reset Value	Description
cntr6clk on page 2-60	0x38	32	RW	0x0	Main PLL Control Register for Counter 6 Clock
cntr7clk on page 2-61	0x3C	32	RW	0x0	Main PLL Control Register for Counter 7 Clock
cntr8clk on page 2-62	0x40	32	RW	0x0	Main PLL Control Register for Counter 8 Clock
outrst on page 2-63	0x60	32	RW	0x0	Main PLL Output Counter Reset Register
outrststat on page 2-64	0x64	32	RO	0x0	Main PLL Output Counter Reset Ack Status Register
nocdiv on page 2-65	0x68	32	RW	0x18020100	NoC Divide Register

clkmgr_mainpllgrp Summary

Base Address: 0xFFD04040

Register Address Offset	Bit Fields															
i_clk_mgr_mainpllgrp																
vco0 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved			bwadjen RW 0x0	bwadj RW 0x1											
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						psrc RW 0x0		Reserved	saten RW 0x1	fasten RW 0x0	regextsel RW 0x0	outreseteall RW 0x0	en RW 0x0	pwrdn RW 0x1	bgpwrdn RW 0x1

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved										denom RW 0x1					
vco1 0x4	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved			numer RW 0x1												
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
en 0x8	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								hmcpllrefclken RW 0x1	s2fuser0clken RW 0x1	cstimerclken RW 0x1	csclken RW 0x1	l4spclken RW 0x1	l4mpclken RW 0x1	l4maiclken RW 0x1	mpuclken RW 0x1
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
ens 0xC	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								hmcpllrefclken RW 0x1	s2fuser0clken RW 0x1	cstimerclken RW 0x1	csclken RW 0x1	l4spclken RW 0x1	l4mpclken RW 0x1	l4maiclken RW 0x1	mpuclken RW 0x1
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
enr 0x10	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								hmcpllrefclken RW 0x1	s2fuser0clken RW 0x1	cstimerclken RW 0x1	csclken RW 0x1	l4spclken RW 0x1	l4mpclken RW 0x1	l4maiclken RW 0x1	mpuclken RW 0x1
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
bypass 0x14	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved										fben RW 0x1	rfen RW 0x1	hmcpllref RW 0x1	s2fuser0 RW 0x1	noc RW 0x1	mpu RW 0x1

Register Address Offset	Bit Fields															
bypasss 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved										fben RW 0x1	rfen RW 0x1	hmcpllref RW 0x1	s2fuser0 RW 0x1	noc RW 0x1	mpu RW 0x1
bypassr 0x1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved										fben RW 0x1	rfen RW 0x1	hmcpllref RW 0x1	s2fuser0 RW 0x1	noc RW 0x1	mpu RW 0x1
mpuclk 0x20	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved													src RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved					cnt RW 0x0										
nocclk 0x24	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved													src RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved					cnt RW 0x0										
cntr2clk 0x28	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved					cnt RW 0x0										
cntr3clk 0x2C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved					cnt RW 0x0										

Register Address Offset	Bit Fields															
cntr4clk 0x30	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved					cnt RW 0x0										
cntr5clk 0x34	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved					cnt RW 0x0										
cntr6clk 0x38	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved					cnt RW 0x0										
cntr7clk 0x3C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved													src RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved					cnt RW 0x0										
cntr8clk 0x40	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved					cnt RW 0x0										
outrst 0x60	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	outreset RW 0x0															
outrststat 0x64	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	outresetack RO 0x0															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
nocdiv 0x68	Reserved			cspdbgclk RW 0x1	cstraceclk RW 0x2		csatclk RW 0x0		Reserved						l4spclk RW 0x2	
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						l4mpclk RW 0x1		Reserved						l4mainclk RW 0x0	

vco0

Module Instance	Base Address	Register Address
i_clk_mgr_mainpllgrp	0xFFD04040	0xFFD04040

Offset: 0x0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved			bwadjen RW 0x0	bwadj RW 0x1											
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						psrc RW 0x0		Reserved	saten RW 0x1	fasten RW 0x0	regextsel RW 0x0	outresetall RW 0x0	en RW 0x0	pwrdn RW 0x1	bgpwrdn RW 0x1

vco0 Fields

Bit	Name	Description	Access	Reset
28	bwadjen	If set to 1, the Loop Bandwidth Adjust value comes from the Loop Bandwidth Adjust field. If set to 0, the Loop Bandwidth Adjust value equals the M field divided by 2 value of the VCO Control Register. The M divided by 2 is the upper 12 bits (12:1) of the M field in the VCO register.	RW	0x0
27:16	bwadj	Provides Loop Bandwidth Adjust value.	RW	0x1
9:8	psrc	Controls the VCO input clock source. **Value** — **Description** 0 — EOSC1 (external oscillator) 1 — cb_intosc_ls_clk (internal low speed oscillator) 2 — f2h_free_clk (FPGA fabric PLL clock reference)	RW	0x0
6	saten	Enables saturation behavior.	RW	0x1
5	fasten	Enables fast locking circuit.	RW	0x0
4	regextsel	If set to '1', the external regulator is selected for the PLL. If set to '0', the internal regulator is slected. It is strongly recommended to select the external regulator while the PLL is not enabled (in reset), and then disable the external regulater once the PLL becomes enabled. Software should simulate-ously update the 'Enable' bit and the 'External Regulator Input Select' in the same write access to the VCO register. When the 'Enable' bit is clear, the 'External Regulator Input Select' should be set, and vice versa. The reset value of this bit is applied on a cold reset; warm reset has no affect on this bit.	RW	0x0

Bit	Name	Description	Access	Reset
3	outresetall	Before releasing Bypass, All Output Counter Reset must be set and cleared by software for correct clock operation. If '1', Reset phase multiplexer and all output counter state. So that after the assertion all the clocks output are start from rising edge align. If '0', phase multiplexer and output counter state not reset and no change to the phase of the clock outputs.	RW	0x0
2	en	If '1', VCO is enabled. If '0', VCO is in reset.	RW	0x0
1	pwrdn	If '1', power down analog circuitry. If '0', analog circuitry not powered down.	RW	0x1
0	bgpwrdn	If '1', powers down bandgap. If '0', bandgap is not power down.	RW	0x1

vco1

Contains settings that control the Main PLL VCO. The VCO1 register contains the numerator and denominator counter settings.

Module Instance	Base Address	Register Address
i_clk_mgr_mainpllgrp	0xFFD04040	0xFFD04044

Offset: 0x4

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved										denom RW 0x1					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			numer RW 0x1												

vco1 Fields

Bit	Name	Description	Access	Reset
21:16	denom	Denominator in VCO output frequency equation. For incremental frequency change, if the new value lead to less than 20% of the frequency change, this value can be changed without resetting the PLL. The Numerator and Denominator can not be changed at the same time for incremental frequency changed.	RW	0x1
12:0	numer	Numerator in VCO output frequency equation. For incremental frequency change, if the new value lead to less than 20% of the frequency change, this value can be changed without resetting the PLL. The Numerator and Denominator can not be changed at the same time for incremental frequency changed.	RW	0x1

en

Contains fields that control clock enables for Main Clocks.
1: The clock is enabled.
0: The clock is disabled.

Module Instance	Base Address	Register Address
i_clk_mgr_mainpllgrp	0xFFD04040	0xFFD04048

Offset: 0x8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								hmcpllrefclken RW 0x1	s2fuser0clken RW 0x1	cstimerclken RW 0x1	csclken RW 0x1	l4spclken RW 0x1	l4mpclken RW 0x1	l4mainclken RW 0x1	mpuclken RW 0x1

en Fields

Bit	Name	Description	Access	Reset
7	hmcpllrefclken	Enables clock hmc_pll_ref_clk output	RW	0x1
6	s2fuser0clken	Enables clock s2f_user0_clk output	RW	0x1
5	cstimerclken	Enables Debug Timer Clock output (cs_timer_clk)	RW	0x1
4	csclken	Enables Debug Clock outputs (cs_at_clk, cs_pdbg_clk, and cs_trace_clk)	RW	0x1
3	l4spclken	Enables clock l4_sp_clk output	RW	0x1
2	l4mpclken	Enables clock l4_mp_clk output	RW	0x1
1	l4mainclken	Enables clock l4_main_clk output	RW	0x1
0	mpuclken	Enable for MPU Clock Group (mpu_clk, mpu_l2ram_clk and mpu_periph_clk).	RW	0x1

ens

Write One to Set correspondng fields in the Enable Register.

Module Instance	Base Address	Register Address
i_clk_mgr_mainpllgrp	0xFFD04040	0xFFD0404C

Offset: 0xC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								hmcpllrefclken RW 0x1	s2fuser0clken RW 0x1	cstimerclken RW 0x1	csclken RW 0x1	l4spclken RW 0x1	l4mpclken RW 0x1	l4mainclken RW 0x1	mpuclken RW 0x1

ens Fields

Bit	Name	Description	Access	Reset
7	hmcpllrefclken	Enables clock hmc_pll_ref_clk output	RW	0x1
6	s2fuser0clken	Enables clock s2f_user0_clk output	RW	0x1
5	cstimerclken	Enables Debug Timer Clock output (cs_timer_clk)	RW	0x1
4	csclken	Enables Debug Clock outputs (cs_at_clk, cs_pdbg_clk and cs_trace_clk)	RW	0x1
3	l4spclken	Enables clock l4_sp_clk output	RW	0x1
2	l4mpclken	Enables clock l4_mp_clk output	RW	0x1
1	l4mainclken	Enables clock l4_main_clk output	RW	0x1
0	mpuclken	Enable for MPU Clock Group (mpu_clk, mpu_l2ram_clk and mpu_periph_clk).	RW	0x1

enr

Write One to Clear corresponding fields in Enable Register.

Module Instance	Base Address	Register Address
i_clk_mgr_mainpllgrp	0xFFD04040	0xFFD04050

Offset: 0x10

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								hmcpllrefclken RW 0x1	s2fuser0clken RW 0x1	cstimerclken RW 0x1	csclken RW 0x1	l4spclken RW 0x1	l4mpclken RW 0x1	l4mainclken RW 0x1	mpuclken RW 0x1

enr Fields

Bit	Name	Description	Access	Reset
7	hmcpllrefclken	Enables clock hmc_pll_ref_clk output	RW	0x1
6	s2fuser0clken	Enables clock s2f_user0_clk output	RW	0x1
5	cstimerclken	Enables Debug Timer Clock output (cs_timer_clk)	RW	0x1
4	csclken	Enables Debug Clock outputs (cs_at_clk, cs_pdbg_clk and cs_trace_clk)	RW	0x1
3	l4spclken	Enables clock l4_sp_clk output	RW	0x1
2	l4mpclken	Enables clock l4_mp_clk output	RW	0x1
1	l4mainclken	Enables clock l4_main_clk output	RW	0x1
0	mpuclken	Enable for MPU Clock Group (mpu_clk, mpu_l2ram_clk and mpu_periph_clk).	RW	0x1

bypass

Contains fields that control bypass for clocks derived from the Main PLL.
1: The clock is bypassed to boot_clk.
0: The clock is derived from the 5:1 active mux.

Module Instance	Base Address	Register Address
i_clk_mgr_mainpllgrp	0xFFD04040	0xFFD04054

Offset: 0x14

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										fben RW 0x1	rfen RW 0x1	hmcpllref RW 0x1	s2fuser0 RW 0x1	noc RW 0x1	mpu RW 0x1

bypass Fields

Bit	Name	Description	Access	Reset
5	fben	If set, the pll_main_fben_clk will be bypassed to the boot_clk. The pll_main_fben_clk is used to synchronously update the Numerator to the Main PLL.	RW	0x1
4	rfen	If set, the pll_main_rfen_clk will be bypassed to the boot_clk. The pll_main_rfen_clk is used to synchronously update the Denominator to the Main PLL.	RW	0x1
3	hmcpllref	If set, the hmc_pll_ref_clk will be bypassed to the boot_clk.	RW	0x1

Bit	Name	Description	Access	Reset
2	s2fuser0	If set, the s2f_user0_clk will be bypassed to the boot_clk.	RW	0x1
1	noc	If set, the NOC clock group will be bypassed to boot_clk.	RW	0x1
0	mpu	If set, the MPU clock group will be bypassed to the boot_clk.	RW	0x1

bypasss

Write One to Set corresponding fields in Bypass Register.

Module Instance	Base Address	Register Address
i_clk_mgr_mainpllgrp	0xFFD04040	0xFFD04058

Offset: 0x18

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										fben RW 0x1	rfen RW 0x1	hmcpllref RW 0x1	s2fuser0 RW 0x1	noc RW 0x1	mpu RW 0x1

bypasss Fields

Bit	Name	Description	Access	Reset
5	fben	If set, the pll_main_fben_clk will be bypassed to the boot_clk. The pll_main_fben_clk is used to synchronously update the Numerator to the Main PLL.	RW	0x1

Bit	Name	Description	Access	Reset
4	rfen	If set, the pll_main_rfen_clk will be bypassed to the boot_clk. The pll_main_rfen_clk is used to synchronously update the Denominator to the Main PLL.	RW	0x1
3	hmcpllref	If set, the hmc_pll_ref_clk will be bypassed to the boot_clk.	RW	0x1
2	s2fuser0	If set, the s2f_user0_clk will be bypassed to the boot_clk.	RW	0x1
1	noc	If set, the NOC clock group will be bypassed to the boot_clk.	RW	0x1
0	mpu	If set, the MPU clock group will be bypassed to the boot_clk.	RW	0x1

bypassr

Write One to Clear corresponding fields in Bypass Register.

Module Instance	Base Address	Register Address
i_clk_mgr_mainpllgrp	0xFFD04040	0xFFD0405C

Offset: 0x1C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										fben RW 0x1	rfen RW 0x1	hmcpllref RW 0x1	s2fuser0 RW 0x1	noc RW 0x1	mpu RW 0x1

bypassr Fields

Bit	Name	Description	Access	Reset
5	fben	If set, the pll_main_fben_clk will be bypassed to the boot_clk. The pll_main_fben_clk is used to synchronously update the Numerator to the Main PLL.	RW	0x1
4	rfen	If set, the pll_main_rfen_clk will be bypassed to the boot_clk. The pll_main_rfen_clk is used to synchronously update the Denominator to the Main PLL.	RW	0x1
3	hmcpllref	If set, the hmc_pll_ref_clk will be bypassed to the boot_clk.	RW	0x1
2	s2fuser0	If set, the s2f_user0_clk will be bypassed to the input clock reference of the Main PLL.	RW	0x1
1	noc	If set, the NOC clock group will be bypassed to the input clock reference of the Main PLL.	RW	0x1
0	mpu	If set, the MPU clock group will be bypassed to the input clock reference of the Main PLL.	RW	0x1

mpuclk

Contains settings that control clock mpu_clk generated from the Main PLL VCO clock.

Module Instance	Base Address	Register Address
i_clk_mgr_mainpllgrp	0xFFD04040	0xFFD04060

Offset: 0x20

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													src RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					cnt RW 0x0										

mpuclk Fields

Bit	Name	Description	Access	Reset
18:16	src	Selects the source for the active 5:1 clock selection when the PLL is not bypassed. **Value** **Description** 0 pll0_mpu_base_clk 1 pll1_mpu_base_clk 2 osc1_clk (external oscillator) 3 cb_intosc_hs_div2_clk (internal high speed oscillator divided by 2) 4 f2h_free_clk (FPGA fabric PLL clock reference)	RW	0x0
10:0	cnt	Divides the VCO/2 frequency by the value+1 in this field.	RW	0x0

nocclk

Contains settings that control clock main_clk generated from the Main PLL VCO clock.

Module Instance	Base Address	Register Address
i_clk_mgr_mainpllgrp	0xFFD04040	0xFFD04064

Offset: 0x24

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													src RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					cnt RW 0x0										

nocclk Fields

Bit	Name	Description	Access	Reset
18:16	src	Selects the source for the active 5:1 clock selection when the PLL is not bypassed. **Value** — **Description** 0 — pll0_noc_base_clk 1 — pll1_noc_base_clk 2 — osc1_clk (external oscillator) 3 — cb_intosc_hs_div2_clk (high speed internal oscillator divided by 2) 4 — f2h_free_clk (FPGA Fabric PLL clock reference)	RW	0x0
10:0	cnt	Divides the VCO frequency by the value+1 in this field.	RW	0x0

cntr2clk

Contains settings that control Counter 2 clock generated from the Main PLL VCO clock.

Module Instance	Base Address	Register Address
i_clk_mgr_mainpllgrp	0xFFD04040	0xFFD04068

Offset: 0x28

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					cnt RW 0x0										

cntr2clk Fields

Bit	Name	Description	Access	Reset
10:0	cnt	Divides the VCO frequency by the value+1 in this field.	RW	0x0

cntr3clk

Contains settings that control Counter 3 clock generated from the Main PLL VCO clock.

Module Instance	Base Address	Register Address
i_clk_mgr_mainpllgrp	0xFFD04040	0xFFD0406C

Offset: 0x2C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					cnt RW 0x0										

cntr3clk Fields

Bit	Name	Description	Access	Reset
10:0	cnt	Divides the VCO frequency by the value+1 in this field.	RW	0x0

cntr4clk

Contains settings that control Counter 4 clock generated from the Main PLL VCO clock.

Module Instance	Base Address	Register Address
i_clk_mgr_mainpllgrp	0xFFD04040	0xFFD04070

Offset: 0x30

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					cnt RW 0x0										

cntr4clk Fields

Bit	Name	Description	Access	Reset
10:0	cnt	Divides the VCO frequency by the value+1 in this field.	RW	0x0

cntr5clk

Contains settings that control Counter 5 clock generated from the Main PLL VCO clock.

Module Instance	Base Address	Register Address
i_clk_mgr_mainpllgrp	0xFFD04040	0xFFD04074

Offset: 0x34

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					cnt RW 0x0										

cntr5clk Fields

Bit	Name	Description	Access	Reset
10:0	cnt	Divides the VCO frequency by the value+1 in this field.	RW	0x0

cntr6clk

Contains settings that control Counter 6 clock generated from the Main PLL VCO clock.

Module Instance	Base Address	Register Address
i_clk_mgr_mainpllgrp	0xFFD04040	0xFFD04078

Offset: 0x38

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					cnt RW 0x0										

cntr6clk Fields

Bit	Name	Description	Access	Reset
10:0	cnt	Divides the VCO frequency by the value+1 in this field.	RW	0x0

cntr7clk

Contains settings that control Counter 7 clock generated from the Main PLL VCO clock.

Module Instance	Base Address	Register Address
i_clk_mgr_mainpllgrp	0xFFD04040	0xFFD0407C

Offset: 0x3C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													src RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					cnt RW 0x0										

cntr7clk Fields

Bit	Name	Description	Access	Reset
18:16	src	Selects the source for the active 5:1 clock selection when the PLL is not bypassed. **Value** — **Description** 0 — pll0_h2f_user0_clk 1 — pll1_h2f_user0_clk 2 — osc1_clk (external oscillator) 3 — cb_intosc_hs_div2_clk (internal high speed oscillator divided by 2) 4 — f2h_free_clk (FPGA fabric PLL clock reference)	RW	0x0
10:0	cnt	Divides the VCO frequency by the value+1 in this field.	RW	0x0

cntr8clk

Contains settings that control Counter 8 clock generated from the Main PLL VCO clock.

Module Instance	Base Address	Register Address
i_clk_mgr_mainpllgrp	0xFFD04040	0xFFD04080

Offset: 0x40

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					cnt RW 0x0										

cntr8clk Fields

Bit	Name	Description	Access	Reset
10:0	cnt	Divides the VCO frequency by the value+1 in this field.	RW	0x0

outrst

Contains settings to assert individual Outreset for all Main PLL Counters.

Module Instance	Base Address	Register Address
i_clk_mgr_mainpllgrp	0xFFD04040	0xFFD040A0

Offset: 0x60

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
outreset RW 0x0															

outrst Fields

Bit	Name	Description	Access	Reset
15:0	outreset	Resets the individual PLL output counter. For software to change the PLL output counter without producing glitches on the respective clock, SW must set the Output Counter Reset Register 'Output Counter Reset' bit. Software then polls the respective Output Counter Reset Acknowledge bit in the Output Counter Reset Ack Status Register. Software then writes the appropriate counter register, and then clears the respective Output Counter Reset bit. LSB 'outreset[0]' corresponds to PLL output clock C0, etc. If set to '1', reset output divider, no clock output from counter. If set to '0', counter is not reset. The reset value of this bit is applied on a cold reset; warm reset has no affect on this bit.	RW	0x0

outrststat

Contains Output Clock Counter Reset acknowledge status.

Module Instance	Base Address	Register Address
i_clk_mgr_mainpllgrp	0xFFD04040	0xFFD040A4

Offset: 0x64

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
outresetack RO 0x0															

outrststat Fields

Bit	Name	Description	Access	Reset
15:0	outresetack	These read only bits per PLL output indicate that the PLL has received the Output Reset Counter request and has gracefully stopped the respective PLL output clock. For software to change the PLL output counter without producing glitches on the respective clock, SW must set the Output Counter Reset Register 'Output Counter Reset' bit. Software then polls the respective Output Counter Reset Acknowledge bit in the Output Counter Reset Ack Status Register. Software then writes the appropriate counter register, and then clears the respective Output Counter Reset bit. The reset value of this bit is applied on a cold reset; warm reset has no affect on this bit. **Value** **Description** 0 IDLE 1 ACK_RECEIVED	RO	0x0

nocdiv

Contains fields that control clock dividers for NoC Clocks.

Module Instance	Base Address	Register Address
i_clk_mgr_mainpllgrp	0xFFD04040	0xFFD040A8

Offset: 0x68

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved			cspdbgclk RW 0x1	cstraceclk RW 0x2		csatclk RW 0x0		Reserved						l4spclk RW 0x2	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						l4mpclk RW 0x1		Reserved						l4mainclk RW 0x0	

nocdiv Fields

Bit	Name	Description	Access	Reset
28	cspdbgclk	The external cs_pdbg_clk divider is specified in this field. This divider is cascaded after the cs_at_clk external divider. **Value** — **Description** 0 — Div1 1 — Div4	RW	0x1
27:26	cstraceclk	The external cs_trace_clk divider is specified in this field. The cs_trace_clk is used by the actual trace interface to the debugger. This divider is cascaded after the cs_at_clk external divider. **Value** — **Description** 0 — Div1 1 — Div2 2 — Div4 3 — Div8	RW	0x2

Bit	Name	Description	Access	Reset
25:24	csatclk	The external cs_at_clk divider is specified in this field. **Value** — **Description** 0 — Div1 1 — Div2 2 — Div4 3 — Div8	RW	0x0
17:16	l4spclk	The external l4_sp_clk divider is specified in this field. **Value** — **Description** 0 — Div1 1 — Div2 2 — Div4 3 — Div8	RW	0x2
9:8	l4mpclk	The external l4_mp_clk divider is specified in this field. **Value** — **Description** 0 — Div1 1 — Div2 2 — Div4 3 — Div8	RW	0x1
1:0	l4mainclk	The external l4_main_clk divider is specified in this field. **Value** — **Description** 0 — Div1 1 — Div2 2 — Div4 3 — Div8	RW	0x0

clkmgr_perpllgrp Address Map

Module Instance	Base Address	End Address
i_clk_mgr_perpllgrp	0xFFD040C0	0xFFD0413F

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
vco0 on page 2-73	0x0	32	RW	0x10043	Peripheral PLL VCO Control Register 0
vco1 on page 2-75	0x4	32	RW	0x10001	Main PLL VCO Control Register
en on page 2-76	0x8	32	RW	0xF7F	Enable Register
ens on page 2-78	0xC	32	RW	0xF7F	Enable Set Register
enr on page 2-79	0x10	32	RW	0xF7F	Enable Reset Register
bypass on page 2-80	0x14	32	RW	0xFF	Bypass Register
bypasss on page 2-82	0x18	32	RW	0xFF	Bypass Set Register
bypassr on page 2-83	0x1C	32	RW	0xFF	Bypass Reset Register
cntr2clk on page 2-84	0x28	32	RW	0x0	Peripheral PLL Control Register for Counter 2 Clock
cntr3clk on page 2-85	0x2C	32	RW	0x0	Peripheral PLL Control Register for Counter 3 Clock
cntr4clk on page 2-86	0x30	32	RW	0x0	Peripheral PLL Control Register for Counter 4 Clock
cntr5clk on page 2-87	0x34	32	RW	0x0	Peripheral PLL Control Register for Counter 5 Clock
cntr6clk on page 2-88	0x38	32	RW	0x0	Peripheral PLL Control Register for Counter 6 Clock

Register	Offset	Width	Access	Reset Value	Description
cntr7clk on page 2-89	0x3C	32	RW	0x0	Peripheral PLL Control Register for Counter 7 Clock
cntr8clk on page 2-90	0x40	32	RW	0x0	Peripheral PLL Control Register for Counter 8 Clock
outrst on page 2-91	0x60	32	RW	0x0	Peripheral PLL Output Counter Reset Register
outrststat on page 2-92	0x64	32	RO	0x0	Peripheral PLL Output Counter Reset Ack Status Register
emacctl on page 2-93	0x68	32	RW	0x0	Main Divide Register
gpiodiv on page 2-95	0x6C	32	RW	0x1	GPIO Divide Register

clkmgr_perpllgrp Summary

Base Address: 0xFFD040C0

Register Address Offset	Bit Fields															
i_clk_mgr_perpllgrp																
vco0 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved			bwadjen RW 0x0	bwadj RW 0x1											
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						psrc RW 0x0		Reserved	saten RW 0x1	fasten RW 0x0	regextsel RW 0x0	outreseteall RW 0x0	en RW 0x0	pwrdn RW 0x1	bgpwrdn RW 0x1

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved										denom RW 0x1					
vco1 0x4	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved			numer RW 0x1												
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
en 0x8	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				qspiclken RW 0x1	nandclken RW 0x1	spimclken RW 0x1	usbclken RW 0x1	Reserved	s2fuser1clken RW 0x1	sdmmcclken RW 0x1	gpiodben RW 0x1	emacptpen RW 0x1	emac2en RW 0x1	emac1en RW 0x1	emac0en RW 0x1
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
ens 0xC	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				qspiclken RW 0x1	nandclken RW 0x1	spimclken RW 0x1	usbclken RW 0x1	Reserved	s2fuser1clken RW 0x1	sdmmcclken RW 0x1	gpiodben RW 0x1	emacptpen RW 0x1	emac2en RW 0x1	emac1en RW 0x1	emac0en RW 0x1
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
enr 0x10	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				qspiclken RW 0x1	nandclken RW 0x1	spimclken RW 0x1	usbclken RW 0x1	Reserved	s2fuser1clken RW 0x1	sdmmcclken RW 0x1	gpiodben RW 0x1	emacptpen RW 0x1	emac2en RW 0x1	emac1en RW 0x1	emac0en RW 0x1
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
bypass 0x14	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								fben RW 0x1	rfen RW 0x1	s2fuser1 RW 0x1	sdmmc RW 0x1	gpiodb RW 0x1	emacptp RW 0x1	emacb RW 0x1	emaca RW 0x1

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
bypasss	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x18	Reserved								fben RW 0x1	rfen RW 0x1	s2fuser1 RW 0x1	sdmmc RW 0x1	gpiodb RW 0x1	emacptp RW 0x1	emacb RW 0x1	emaca RW 0x1
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
bypassr	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x1C	Reserved								fben RW 0x1	rfen RW 0x1	s2fuser1 RW 0x1	sdmmc RW 0x1	gpiodb RW 0x1	emacptp RW 0x1	emacb RW 0x1	emaca RW 0x1
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved													src RW 0x0		
cntr2clk	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x28	Reserved					cnt RW 0x0										
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved													src RW 0x0		
cntr3clk	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x2C	Reserved					cnt RW 0x0										
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved													src RW 0x0		
cntr4clk	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x30	Reserved					cnt RW 0x0										
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved													src RW 0x0		
cntr5clk	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x34	Reserved					cnt RW 0x0										

Register Address Offset	Bit Fields															
cntr6clk 0x38	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved													src RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved					cnt RW 0x0										
cntr7clk 0x3C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved					cnt RW 0x0										
cntr8clk 0x40	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved													src RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved					cnt RW 0x0										
outrst 0x60	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	outreset RW 0x0															
outrststat 0x64	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	outresetack RO 0x0															
emacctl 0x68	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved			emac2sel RW 0x0	emac1sel RW 0x0	emac0sel RW 0x0	Reserved									
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
gpiodiv 0x6C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	gpiodbclk RW 0x1															

vco0

Contains settings that control the Peripheral PLL VCO. VCO0 register contains signals required for PLL reset and power down.

Module Instance	Base Address	Register Address
i_clk_mgr_perpllgrp	0xFFD040C0	0xFFD040C0

Offset: 0x0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved			bwadjen RW 0x0	bwadj RW 0x1											
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						psrc RW 0x0		Reserved	saten RW 0x1	fasten RW 0x0	regextsel RW 0x0	outresetall RW 0x0	en RW 0x0	pwrdn RW 0x1	bgpwrdn RW 0x1

vco0 Fields

Bit	Name	Description	Access	Reset
28	bwadjen	If set to 1, the Loop Bandwidth Adjust value comes from the Loop Bandwidth Adjust field. If set to 0, the Loop Bandwidth Adjust value equals the M field divided by 2 value of the VCO Control Register. The M divided by 2 is the upper 12 bits (12:1) of the M field in the VCO register.	RW	0x0
27:16	bwadj	Provides Loop Bandwidth Adjust value.	RW	0x1
9:8	psrc	Controls the VCO input clock source. Value: Description 0: EOSC1 1: INTOSC 2: F2S 3: MAIN	RW	0x0
6	saten	Enables saturation behavior.	RW	0x1
5	fasten	Enables fast locking circuit.	RW	0x0
4	regextsel	If set to '1', the external regulator is selected for the PLL. If set to '0', the internal regulator is slected. It is strongly recommended to select the external regulator while the PLL is not enabled (in reset), and then disable the external regulater once the PLL becomes enabled. Software should simulate-ously update the 'Enable' bit and the 'External Regulator Input Select' in the same write access to the VCO register. When the 'Enable' bit is clear, the 'External Regulator Input Select' should be set, and vice versa. The reset value of this bit is applied on a cold reset; warm reset has no affect on this bit.	RW	0x0

Bit	Name	Description	Access	Reset
3	outresetall	Before releasing Bypass, All Output Counter Reset must be set and cleared by software for correct clock operation. If '1', Reset phase multiplexer and all output counter state. So that after the assertion all the clocks output are start from rising edge align. If '0', phase multiplexer and output counter state not reset and no change to the phase of the clock outputs.	RW	0x0
2	en	If '1', VCO is enabled. If '0', VCO is in reset.	RW	0x0
1	pwrdn	If '1', power down analog circuitry. If '0', analog circuitry not powered down.	RW	0x1
0	bgpwrdn	If '1', powers down bandgap. If '0', bandgap is not power down.	RW	0x1

vco1

Contains settings that control the Peripheral PLL VCO. The VCO1 register contains the numerator and denominator counter settings.

Module Instance	Base Address	Register Address
i_clk_mgr_perpllgrp	0xFFD040C0	0xFFD040C4

Offset: 0x4

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved										denom RW 0x1					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			numer RW 0x1												

vco1 Fields

Bit	Name	Description	Access	Reset
21:16	denom	Denominator in VCO output frequency equation. For incremental frequency change, if the new value lead to less than 20% of the frequency change, this value can be changed without resetting the PLL. The Numerator and Denominator can not be changed at the same time for incremental frequency changed.	RW	0x1
12:0	numer	Numerator in VCO output frequency equation. For incremental frequency change, if the new value lead to less than 20% of the frequency change, this value can be changed without resetting the PLL. The Numerator and Denominator can not be changed at the same time for incremental frequency changed.	RW	0x1

en

Contains fields that control clock enables for clocks derived from the Peripheral PLL.
1: The clock is enabled.
0: The clock is disabled.

Module Instance	Base Address	Register Address
i_clk_mgr_perpllgrp	0xFFD040C0	0xFFD040C8

Offset: 0x8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				qspiclken RW 0x1	nandclken RW 0x1	spimclken RW 0x1	usbclken RW 0x1	Reserved	s2fuser1clken RW 0x1	sdmmcclken RW 0x1	gpiodben RW 0x1	emacptpen RW 0x1	emac2en RW 0x1	emac1en RW 0x1	emac0en RW 0x1

en Fields

Bit	Name	Description	Access	Reset
11	qspiclken	Enables QSPI peripheral clock. This enable goes outside of the Clock Manger to the QSPI directly.	RW	0x1
10	nandclken	Enables NAND peripheral clock. This enable goes outside of the Clock Manger to the NAND directly.	RW	0x1
9	spimclken	Enables SPI Master peripheral clock. This enable goes outside of the Clock Manger to the SPIM directly.	RW	0x1
8	usbclken	Enables USB peripheral clock. This enable goes outside of the Clock Manger to the USB directly.	RW	0x1
6	s2fuser1clken	Enables clock s2f_user1_clk output	RW	0x1
5	sdmmcclken	Enables SDMMC peripheral clock. This enable goes outside of the Clock Manger to the SDMMC directly.	RW	0x1
4	gpiodben	Enables clock gpio_db_clk output	RW	0x1
3	emacptpen	Enables clock emac_ptp_clk output	RW	0x1
2	emac2en	Enables clock emac2_clk output	RW	0x1
1	emac1en	Enables clock emac1_clk output	RW	0x1

Bit	Name	Description	Access	Reset
0	emac0en	Enables clock emac0_clk output	RW	0x1

ens

Write One to Set corresonding fields in Enable Register.

Module Instance	Base Address	Register Address
i_clk_mgr_perpllgrp	0xFFD040C0	0xFFD040CC

Offset: 0xC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				qspiclken RW 0x1	nandclken RW 0x1	spimclken RW 0x1	usbclken RW 0x1	Reserved	s2fuser1clken RW 0x1	sdmmcclken RW 0x1	gpiodben RW 0x1	emacptpen RW 0x1	emac2en RW 0x1	emac1en RW 0x1	emac0en RW 0x1

ens Fields

Bit	Name	Description	Access	Reset
11	qspiclken	Enables QSPI peripheral clock. This enable goes outside of the Clock Manger to the QSPI directly.	RW	0x1
10	nandclken	Enables NAND peripheral clock. This enable goes outside of the Clock Manger to the NAND directly.	RW	0x1
9	spimclken	Enables SPI Master peripheral clock. This enable goes outside of the Clock Manger to the SPIM directly.	RW	0x1

Bit	Name	Description	Access	Reset
8	usbclken	Enables USB peripheral clock. This enable goes outside of the Clock Manger to the USB directly.	RW	0x1
6	s2fuser1clken	Enables clock s2f_user1_clk output	RW	0x1
5	sdmmcclken	Enables SDMMC peripheral clock. This enable goes outside of the Clock Manger to the SDMMC directly.	RW	0x1
4	gpiodben	Enables clock gpio_db_clk output	RW	0x1
3	emacptpen	Enables clock emac_ptp_clk output	RW	0x1
2	emac2en	Enables clock emac2_clk output	RW	0x1
1	emac1en	Enables clock emac1_clk output	RW	0x1
0	emac0en	Enables clock emac0_clk output	RW	0x1

enr

Write One to Clear corresponding fields in Enable Register.

Module Instance	Base Address	Register Address
i_clk_mgr_perpllgrp	0xFFD040C0	0xFFD040D0

Offset: 0x10

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				qspiclken RW 0x1	nandclken RW 0x1	spimclken RW 0x1	usbclken RW 0x1	Reserved	s2fuser1clken RW 0x1	sdmmcclken RW 0x1	gpiodben RW 0x1	emacptpen RW 0x1	emac2en RW 0x1	emac1en RW 0x1	emac0en RW 0x1

enr Fields

Bit	Name	Description	Access	Reset
11	qspiclken	Enables QSPI peripheral clock. This enable goes outside of the Clock Manger to the QSPI directly.	RW	0x1
10	nandclken	Enables NAND peripheral clock. This enable goes outside of the Clock Manger to the NAND directly.	RW	0x1
9	spimclken	Enables SPI Master peripheral clock. This enable goes outside of the Clock Manger to the SPIM directly.	RW	0x1
8	usbclken	Enables USB peripheral clock. This enable goes outside of the Clock Manger to the USB directly.	RW	0x1
6	s2fuser1clken	Enables clock s2f_user1_clk output	RW	0x1
5	sdmmcclken	Enables SDMMC peripheral clock. This enable goes outside of the Clock Manger to the SDMMC directly.	RW	0x1
4	gpiodben	Enables clock gpio_db_clk output	RW	0x1
3	emacptpen	Enables clock emac_ptp_clk output	RW	0x1
2	emac2en	Enables clock emac2_clk output	RW	0x1
1	emac1en	Enables clock emac1_clk output	RW	0x1
0	emac0en	Enables clock emac0_clk output	RW	0x1

bypass

Contains fields that control bypass for clocks derived from the Peripheral PLL.
1: The clock is bypassed.
0: The clock is derived from the 5:1 active mux.

Module Instance	Base Address	Register Address
i_clk_mgr_perpllgrp	0xFFD040C0	0xFFD040D4

Offset: 0x14

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								fben RW 0x1	rfen RW 0x1	s2fuser1 RW 0x1	sdmmc RW 0x1	gpiodb RW 0x1	emacptp RW 0x1	emacb RW 0x1	emaca RW 0x1

bypass Fields

Bit	Name	Description	Access	Reset
7	fben	If set, the pll_main_fben_clk will be bypassed to the boot_clk. The pll_main_fben_clk is used to synchronously update the Numerator to the Main PLL.	RW	0x1
6	rfen	If set, the pll_peri_rfen_clk will be bypassed to the boot_clk. The pll_peri_rfen_clk is used to synchronously update the Denominator to the Peripheral PLL.	RW	0x1
5	s2fuser1	If set, the s2f_user1_clk will be bypassed to the boot_clk.	RW	0x1
4	sdmmc	If set, the sdmmc_clk will be bypassed to the boot_clk.	RW	0x1
3	gpiodb	If set, the gpio_db_clk will be bypassed to the boot_clk.	RW	0x1
2	emacptp	If set, the emac_ptp_clk will be bypassed to the boot_clk.	RW	0x1
1	emacb	If set, the emacb_free_clk will be bypassed to the boot_clk.	RW	0x1
0	emaca	If set, the emaca_free_clk will be bypassed to the boot_clk.	RW	0x1

bypasss

Write One to Set corresponding fields in Bypass Register.

Module Instance	Base Address	Register Address
i_clk_mgr_perpllgrp	0xFFD040C0	0xFFD040D8

Offset: 0x18

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								fben RW 0x1	rfen RW 0x1	s2fuser1 RW 0x1	sdmmc RW 0x1	gpiodb RW 0x1	emacptp RW 0x1	emacb RW 0x1	emaca RW 0x1

bypasss Fields

Bit	Name	Description	Access	Reset
7	fben	If set, the pll_main_fben_clk will be bypassed and the boot_clk will be used. The pll_main_fben_clk is used to synchronously update the Numerator to the Main PLL.	RW	0x1
6	rfen	If set, the pll_peri_rfen_clk will be bypassed and the boot_clk will be used. The pll_peri_rfen_clk is used to synchronously update the Denominator to the Peripheral PLL.	RW	0x1
5	s2fuser1	If set, the s2f_user1_clk will be bypassed and the boot_clk will be used.	RW	0x1

Bit	Name	Description	Access	Reset
4	sdmmc	If set, the sdmmc_clk will be bypassed and the boot_clk will be used.	RW	0x1
3	gpiodb	If set, the gpio_db_clk will be bypassed and the boot_clk will be used.	RW	0x1
2	emacptp	If set, the emac_ptp_clk will be bypassed and the boot_clk will be used.	RW	0x1
1	emacb	If set, the emacb_free_clk will be bypassed and the boot_clk will be used.	RW	0x1
0	emaca	If set, the emaca_free_clk will be bypassed and the boot_clk will be used.	RW	0x1

bypassr

Write One to Clear corresponding fields in Bypass Register.

Module Instance	Base Address	Register Address
i_clk_mgr_perpllgrp	0xFFD040C0	0xFFD040DC

Offset: 0x1C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								fben RW 0x1	rfen RW 0x1	s2fuser1 RW 0x1	sdmmc RW 0x1	gpiodb RW 0x1	emacptp RW 0x1	emacb RW 0x1	emaca RW 0x1

bypassr Fields

Bit	Name	Description	Access	Reset
7	fben	If set, the pll_main_fben_clk will be bypassed to the boot_clk. The pll_main_fben_clk is used to synchronously update the Numerator to the Main PLL.	RW	0x1
6	rfen	If set, the pll_peri_rfen_clk will be bypassed to the boot_clk. The pll_peri_rfen_clk is used to synchronously update the Denominator to the Peripheral PLL.	RW	0x1
5	s2fuser1	If set, the s2f_user1_clk will be bypassed to the input clock reference of the Peripheral PLL.	RW	0x1
4	sdmmc	If set, the sdmmc_clk will be bypassed to the input clock reference of the Peripheral PLL.	RW	0x1
3	gpiodb	If set, the gpio_db_clk will be bypassed to the input clock reference of the Peripheral PLL.	RW	0x1
2	emacptp	If set, the emac_ptp_clk will be bypassed to the input clock reference of the Peripheral PLL.	RW	0x1
1	emacb	If set, the emacb_free_clk will be bypassed to the input clock reference of the Peripheral PLL.	RW	0x1
0	emaca	If set, the emaca_free_clk will be bypassed to the input clock reference of the Peripheral PLL.	RW	0x1

cntr2clk

Contains settings that control Counter 2 clock generated from the Peripheral PLL VCO clock.

Module Instance	Base Address	Register Address
i_clk_mgr_perpllgrp	0xFFD040C0	0xFFD040E8

Offset: 0x28

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													src RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					cnt RW 0x0										

cntr2clk Fields

Bit	Name	Description	Access	Reset
18:16	src	Selects the source for the active 5:1 clock selection when the PLL is not bypassed. **Value** — **Description** 0 — pll0_emaca_base_clk 1 — pll1_emaca_base_clk 2 — osc1_clk (external oscillator) 3 — cb_intosc_hs_div2_clk (internal high speed oscillator divided by 2) 4 — f2h_free_clk (FPGA fabric PLL clock reference)	RW	0x0
10:0	cnt	Divides the VCO frequency by the value+1 in this field.	RW	0x0

cntr3clk

Contains settings that control Counter 3 clock generated from the Peripheral PLL VCO clock.

Module Instance	Base Address	Register Address
i_clk_mgr_perpllgrp	0xFFD040C0	0xFFD040EC

Offset: 0x2C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													src RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					cnt RW 0x0										

cntr3clk Fields

Bit	Name	Description	Access	Reset
18:16	src	Selects the source for the active 5:1 clock selection when the PLL is not bypassed. **Value** **Description** 0 pll0_emacb_base_clk 1 pll1_emacb_base_clk 2 osc1_clk (external oscillator) 3 cb_intosc_hs_div2_clk (internal high speed oscillator divided by 2) 4 f2h_free_clk (FPGA fabric PLL clock reference)	RW	0x0
10:0	cnt	Divides the VCO frequency by the value+1 in this field.	RW	0x0

cntr4clk

Contains settings that control Counter 4 clock generated from the Peripheral PLL VCO clock.

Module Instance	Base Address	Register Address
i_clk_mgr_perpllgrp	0xFFD040C0	0xFFD040F0

Offset: 0x30

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													src RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					cnt RW 0x0										

cntr4clk Fields

Bit	Name	Description	Access	Reset
18:16	src	Selects the source for the active 5:1 clock selection when the PLL is not bypassed. **Value** — **Description** 0 — pll0_emac_ptp_clk 1 — pll1_emac_ptp_clk 2 — osc1_clk (external oscillator) 3 — cb_intosc_hs_div2_clk (internal high speed oscillator divided by 2) 4 — f2h_free_clk (FPGA fabric PLL clock reference)	RW	0x0
10:0	cnt	Divides the VCO frequency by the value+1 in this field.	RW	0x0

cntr5clk

Contains settings that control Counter 5 clock generated from the Peripheral PLL VCO clock.

Module Instance	Base Address	Register Address
i_clk_mgr_perpllgrp	0xFFD040C0	0xFFD040F4

Offset: 0x34

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													src RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					cnt RW 0x0										

cntr5clk Fields

Bit	Name	Description	Access	Reset
18:16	src	Selects the source for the active 5:1 clock selection when the PLL is not bypassed. **Value** — **Description** 0 — pll0_gpio_clk 1 — pll1_gpio_clk 2 — osc1_clk (external oscillator) 3 — cb_intosc_hs_div2_clk (internal high speed oscillator divided by 2) 4 — f2h_free_clk (FPGA fabric PLL clock reference)	RW	0x0
10:0	cnt	Divides the VCO frequency by the value+1 in this field.	RW	0x0

cntr6clk

Contains settings that control Counter 6 clock generated from the Peripheral PLL VCO clock.

Module Instance	Base Address	Register Address
i_clk_mgr_perpllgrp	0xFFD040C0	0xFFD040F8

Offset: 0x38

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													src RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					cnt RW 0x0										

cntr6clk Fields

Bit	Name	Description	Access	Reset
18:16	src	Selects the source for the active 5:1 clock selection when the PLL is not bypassed. **Value** — **Description** 0 — pll0_sdmmc_clk 1 — pll1_sdmmc_clk 2 — osc1_clk (external oscillator) 3 — cb_intosc_hs_div2_clk (internal high speed oscillator divided by 2) 4 — f2h_free_clk (FPGA fabric PLL clock reference)	RW	0x0
10:0	cnt	Divides the VCO frequency by the value+1 in this field.	RW	0x0

cntr7clk

Contains settings that control Counter 7 clock generated from the Peripheral PLL VCO clock.

Module Instance	Base Address	Register Address
i_clk_mgr_perpllgrp	0xFFD040C0	0xFFD040FC

Offset: 0x3C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					cnt RW 0x0										

cntr7clk Fields

Bit	Name	Description	Access	Reset
10:0	cnt	Divides the VCO frequency by the value+1 in this field.	RW	0x0

cntr8clk

Contains settings that control Counter 8 clock generated from the Peripheral PLL VCO clock.

Module Instance	Base Address	Register Address
i_clk_mgr_perpllgrp	0xFFD040C0	0xFFD04100

Offset: 0x40

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													src RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					cnt RW 0x0										

cntr8clk Fields

Bit	Name	Description	Access	Reset
18:16	src	Selects the source for the active 5:1 clock selection when the PLL is not bypassed. **Value** **Description** 0 pll0_h2f_user1_clk 1 pll1_h2f_user1_clk 2 osc1_clk (external oscillator) 3 cb_intosc_hs_div2_clk (internal high speed oscillator divided by 2) 4 f2h_free_clk (FPGA fabric PLL clock reference)	RW	0x0
10:0	cnt	Divides the VCO frequency by the value+1 in this field.	RW	0x0

outrst

Contains settings to assert individual Outreset for all Peripheral PLL Counters.

Module Instance	Base Address	Register Address
i_clk_mgr_perpllgrp	0xFFD040C0	0xFFD04120

Offset: 0x60

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
outreset RW 0x0															

outrst Fields

Bit	Name	Description	Access	Reset
15:0	outreset	Resets the individual PLL output counter. For software to change the PLL output counter without producing glitches on the respective clock, SW must set the Output Counter Reset Register 'Output Counter Reset' bit. Software then polls the respective Output Counter Reset Acknowledge bit in the Output Counter Reset Ack Status Register. Software then writes the appropriate counter register, and then clears the respective Output Counter Reset bit. LSB 'outreset[0]' corresponds to PLL output clock C0, etc. If set to '1', reset output divider, no clock output from counter. If set to '0', counter is not reset. The reset value of this bit is applied on a cold reset; warm reset has no affect on this bit.	RW	0x0

outrststat

Contains Output Clock Counter Reset acknowledge status.

Module Instance	Base Address	Register Address
i_clk_mgr_perpllgrp	0xFFD040C0	0xFFD04124

Offset: 0x64

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
outresetack RO 0x0															

outrststat Fields

Bit	Name	Description	Access	Reset
15:0	outresetack	These read only bits per PLL output indicate that the PLL has received the Output Reset Counter request and has gracefully stopped the respective PLL output clock. For software to change the PLL output counter without producing glitches on the respective clock, SW must set the Output Counter Reset Register 'Output Counter Reset' bit. Software then polls the respective Output Counter Reset Acknowledge bit in the Output Counter Reset Ack Status Register. Software then writes the appropriate counter register, and then clears the respective Output Counter Reset bit. The reset value of this bit is applied on a cold reset; warm reset has no affect on this bit. **Value** **Description** 0 IDLE 1 ACK_RECEIVED	RO	0x0

emacctl

Contains fields that control clock dividers for main clocks derived from the Main PLL

Module Instance	Base Address	Register Address
i_clk_mgr_perpllgrp	0xFFD040C0	0xFFD04128

Offset: 0x68

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved			emac2sel RW 0x0	emac1sel RW 0x0	emac0sel RW 0x0	Reserved									
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															

emacctl Fields

Bit	Name	Description	Access	Reset
28	emac2sel	Selects the source for emac2_clk as either emaca_free_clk or emacb_free_clk. Value: Description 0: EMACA 1: EMACB	RW	0x0
27	emac1sel	Selects the source for emac1_clk as either emaca_free_clk or emacb_free_clk. Value: Description 0: EMACA 1: EMACB	RW	0x0

Bit	Name	Description	Access	Reset
26	emac0sel	Selects the source for emac0_clk as either emaca_free_clk or emacb_free_clk. Value: Description 0: EMACA 1: EMACB	RW	0x0

gpiodiv

Contains a field that controls the clock divider for the GPIO De-bounce clock.

Module Instance	Base Address	Register Address
i_clk_mgr_perpllgrp	0xFFD040C0	0xFFD0412C

Offset: 0x6C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
gpiodbclk RW 0x1															

gpiodiv Fields

Bit	Name	Description	Access	Reset
15:0	gpiodbclk	The gpio_db_clk is divided down from the periph_base_clk by the value plus one specified in this field. The value 0 (divide by 1) is illegal. A value of 1 indicates divide by 2, 2 divide by 3, etc.	RW	0x1

clkmgr_alteragrp Address Map

Module Instance	Base Address	End Address
i_clk_mgr_alteragrp	0xFFD04140	0xFFD04FFF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
mpuclk on page 2-97	0x0	32	RW	0x10001	MPU Internal PLL Counters
nocclk on page 2-98	0x4	32	RW	0x30003	NOC Internal PLL Counters

clkmgr_alteragrp Summary

Base Address: 0xFFD04140

Register Address Offset	Bit Fields															
i_clk_mgr_alteragrp																
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved					pericnt										
mpuclk						RW 0x1										
0x0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved					maincnt										
						RW 0x1										
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved					pericnt										
nocclk						RW 0x3										
0x4	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved					maincnt										
						RW 0x3										

mpuclk

Contains internal counter setting for MPU Clock groups for both PLLs.

Module Instance	Base Address	Register Address
i_clk_mgr_alteragrp	0xFFD04140	0xFFD04140

Offset: 0x0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved					pericnt RW 0x1										
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					maincnt RW 0x1										

mpuclk Fields

Bit	Name	Description	Access	Reset
26:16	pericnt	Divides the VCO frequency by the value+1 in this field. This field loads the internal counter in the Main PLL for the Main Clock Group. **Note:** Intel recommends that you do not alter the value of this register. Changing the value at run time can cause the system to become unstable.	RW	0x1
10:0	maincnt	Divides the VCO frequency by the value+1 in this field. This field loads the internal counter in the Main PLL for the Main Clock Group. **Note:** Intel recommends that you do not alter the value of this register. Changing the value at run time can cause the system to become unstable.	RW	0x1

nocclk

Contains settings that control clock main_clk generated from the Main PLL VCO clock.

Module Instance	Base Address	Register Address
i_clk_mgr_alteragrp	0xFFD04140	0xFFD04144

Offset: 0x4

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved					pericnt RW 0x3										
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					maincnt RW 0x3										

nocclk Fields

Bit	Name	Description	Access	Reset
26:16	pericnt	Divides the VCO frequency by the value+1 in this field. This field loads the internal counter in the NOC PLL for the NOC Clock Group.	RW	0x3
10:0	maincnt	Divides the VCO frequency by the value+1 in this field. This field loads the internal counter in the NOC PLL for the NOC Clock Group.	RW	0x3

Document Revision History

Table 2-8: Document Revision History

Date	Version	Changes
October 2016	2016.10.28	• Added `mpuclk` register (offset 0x0) to i_clk_mgr_alteragrp
May 2016	2016.05.27	• Removed references to `f2h_emac*_ap_clk` in the *Arria 10 Top Level Clocks* table. The Ethernet application interface (also called the switch interface) is not supported. • Removed `clk9cntr` (offset 0x44) and `clk15cntr` (offset 0x5C) registers from the i_clk_mgr_mainpllgrp • Removed `clk9cntr` register (offset 0x44) from the i_clk_mgr_perpllgrp • Clarified the `src` fields of the `clk*cntr` registers in the i_clk_mgr_mainpllgrp and i_clk_mgr_perpllgrp Removed references to PLL counter outputs C9-C15 from the following topics: • Boot Clock • PLLs • FREF, FVCO, and FOUT Equations
May 2016	2016.05.03	Added a section titled "L4 Peripheral Clocks".
November 2015	2015.11.02	Updated Sections: • Clock Manager Block Diagram and System Integration • PLL Integration • Software Sequenced Clocks • Clock Gating • Boot Clock Added Sections: • Updating PLL Settings without a System Reset • MPU Clock Scaling Updates: • Removed references to PLL output C9 used as HMC PLL reference clock.
May 2015	2015.05.04	• Updated Block Diagram with HMC block • Added Arria 10 Top Level Clocks table • Updated PLL Integration in Clock Manager figure • Added Address Map and Register Descriptions

Date	Version	Changes
December 2014	2014.12.15	Clock Manager Block Diagram. Updated mux output route. NOC clock added. Peripheral Clocks block update C15 input for PLL1 has been removed throughout document.
August 2014	2014.08.18	Initial release.

Reset Manager 3

2016.10.28

a10_5v4 Subscribe Send Feedback

The reset manager generates module reset signals based on reset requests from the various sources in the HPS and FPGA fabric, and software writing to the module-reset control registers. The reset manager ensures that a reset request from the FPGA fabric can occur only after the FPGA portion of the system-on-a-chip (SoC) device is configured.

The HPS contains multiple reset domains. Each reset domain can be reset independently. A reset may be initiated externally, internally or through software.

The Reset domains and sequences include security features. The security manager works with Power On Reset (POR) and brings the reset manager out of reset only when the secure fuses have been loaded and validated. Once this process is complete, the reset manager brings the rest of the HPS out of reset.

Table 3-1: HPS Reset Domains

Domain Name	Domain Logic	Module Master	Description
POR	Power on Reset. The entire HPS is reset.	Security Manager	After POR, the security manager comes out of reset and validates the security fuses. After validation, the reset manager is released from reset and proceeds to perform a cold reset on the HPS.
System Cold	Cold Reset. All of the HPS except security manager and POR fuse logic.	Reset Manager	Using the known security state, the HPS is placed in the default state, allowing software to boot. Cold reset is triggered by POR as well as other sources.

Domain Name	Domain Logic	Module Master	Description
System Warm	System Warm (SW) is a warm reset. All of the HPS except security manager, POR, Fuse Logic, and test access port (TAP), and Debug domains are reset.	Reset Manager	Used to recover system from a non-responsive condition. Resets a subset of the HPS state reset by a cold reset. Only affects the system reset domain, which allows debugging (including trace) to operate through the warm reset. It is possible to mask a warm reset for some modules such that they are not affected.
Debug	All debug logic including most of the DAP, CoreSight™ components connected to the debug peripheral bus, trace, the microprocessor unit (MPU) subsystem, and the FPGA fabric.	Reset Manager	May be asserted by cold reset, the debugger or software. Used to recover debug logic from a non-responsive condition.
TAP	JTAG test access port (TAP) controller, which is used by the debug access port (DAP).	Reset Manager	Asserted by cold reset or by Software.
RAM Clear	The on-chip memories are cleared	Reset Manager	Memories may be cleared on cold or warm as indicated by the corresponding bits in the `ramstat` register.

The HPS supports the following reset types:

- System cold reset
  - Used to ensure the HPS is placed in a default state sufficient for software to boot
  - Triggered by a power-on reset and other sources
  - Resets all HPS logic that can be reset
  - Affects all reset domains
- System warm reset
  - Used to recover system from a non-responsive condition
  - Does not reset the Debug or TAP reset domain, allowing debug functions including trace to operate through out a warm reset
  - Masks allow software to exclude modules from warm reset. Exceptions for masks:
    - To maintain security, there is no mask for security manager
    - The MPU cannot be masked
  - RAM Clearing domain (if not masked) logic is reset
- Debug reset
  - Used to recover debug logic from a non-responsive condition
  - Only affects the debug reset domain
- TAP Reset
  - JTAG TAP controller is reset
  - Software may trigger TAP reset

Related Information

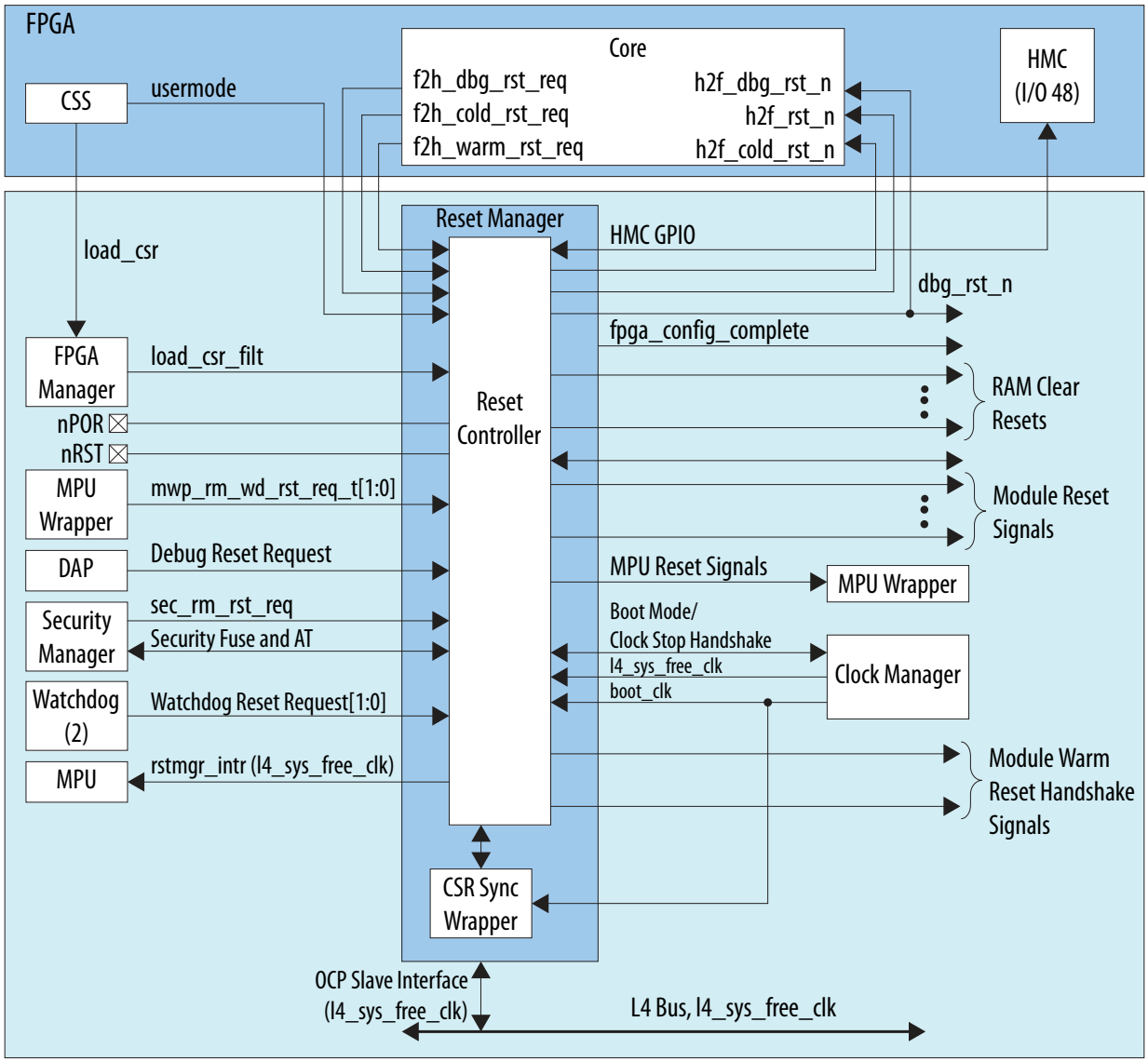
SoC Security on page 6-1

Reset Manager Block Diagram and System Integration

The reset manager accepts reset requests from the security manager, FPGA control block, FPGA core, modules in the HPS, and reset pins and generates module reset signals. The reset manager receives security status from the security manager. The reset manager also has warm reset handshaking signals to support system reset behavior, clock manager interfaces, and the Anti-Tamper interface with security manager. The Boot clock (`boot_clk`) is used as the default clock for both cold or warm reset. When the device is configured to operate in secure mode, `boot_clk` is `cb_intosc_hs_div2_clk`. When the device is not in secure mode, `boot_clk` is EOSC1.

The following figure shows a block diagram of the reset manager in the SoC device. For clarity, reset-related handshaking signals to other HPS modules and to the clock manager module are omitted.

Figure 3-1: Reset Manager Block Diagram



Reset Controller

In secure mode, all signals are synchronous to `boot_clk`. In non-secure mode, all signals are synchronous to `osc1_clk`. The following table lists the reset sources external to the HPS.

Table 3-2: HPS External Reset Sources

Source	Description
`f2h_cold_rst_req_n`	Cold reset request from FPGA fabric (active low)
`f2h_warm_rst_req_n`	Warm reset request from FPGA fabric (active low)

Source	Description
f2h_dbg_rst_req_n	Debug reset request from FPGA fabric (active low)
load_csr_filt	Cold-only reset from FPGA control block (CB)
nPOR	Power-on reset pin (active low)
nRST	Warm reset pin (active low)

Table 3-3: HPS External Reset Outputs

Source	Description
h2f_cold_rst_n	Cold-only reset to FPGA fabric (active low)
h2f_rst_n	Cold or warm reset to FPGA fabric (active low)
h2f_dbg_rst_n	Debug reset (dbg_rst_n) to FPGA fabric (active low)

The reset controller performs the following functions:

- Accepts reset requests from the FPGA control block (CB), FPGA fabric, modules in the HPS, and reset pins
- Generates an individual reset signal for each module instance for all modules in the HPS
- Provides reset handshaking signals to support system reset behavior

The reset controller generates module reset signals from external reset requests and internal reset requests. External reset requests originate from sources external to the reset manager. Internal reset requests originate from control registers in the reset manager.

The reset controller supports the following cold reset requests:

- Security manager reset
- Cold reset request pin (nPOR)
- FPGA fabric
- FPGA CB
- Software cold reset request bit (swcoldrstreq) of the control register (ctrl)

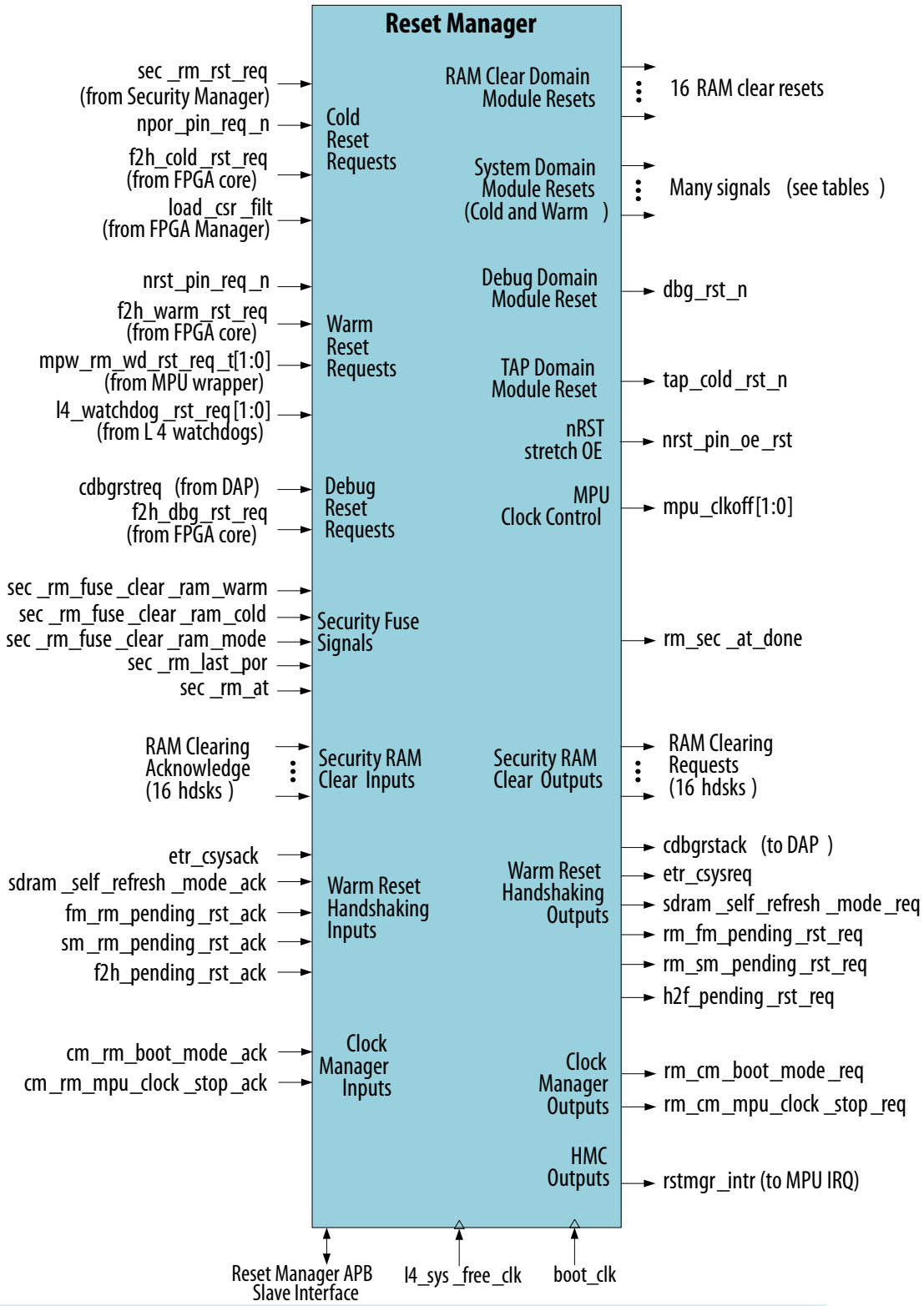
The reset controller supports the following warm reset requests:

- Warm reset request pin (nRST)
- FPGA fabric
- Software warm reset request bit (swwarmrstreq) of the ctrl register
- MPU watchdog reset requests for CPU0 and CPU1
- System watchdog timer 0 and 1 reset requests

The reset controller supports the following debug reset requests:

- CDBGRSTREQ from DAP
- FPGA fabric

Figure 3-2: Reset Controller Signals



Security Reset Functions

When secure mode is enabled on the device, reset manager performs some additional functions. To avoid unsecure snooping of existing RAM contents, the HPS can be configured to clear all RAM, both general-purpose RAM and dedicated RAM that is integrated into modules. The list of RAMs on the HPS is:

- Onchip RAM
- USB0
- USB1
- SDMMC
- EMAC0 - EMAC2 RX and TX buffers
- DMA
- NAND Read, Write and ECC RAMs
- QSPI
- MPU RAM

RAMs can be cleared on cold reset, warm reset or both as there are separate fuses that control behavior for cold and warm resets. When the corresponding fuse is blown, all RAMs are cleared for the indicated reset type. If an additional fuse is blown, the various RAM contents are cleared sequentially; otherwise, the contents of all of the RAMs are cleared simultaneously. Even if all security RAM clearing is disabled, the reset manager always invalidates the MPU L1 Cache.

The HPS also supports an Anti-Tamper interface from the FPGA fabric to security manager. If an Anti-Tamper event occurs, the HPS clears all RAMs and becomes unresponsive. The HPS only recovers from this state by a POR. Anti-Tamper logic, such as voltage detection, may reside in the FPGA.

Module Reset Signals

The following tables list the module reset signals. The module reset signals are organized in groups for the MPU, peripherals, bridges.

In the following tables, columns marked for Cold Reset, Warm Reset, and Debug Reset denote reset signals asserted by each type of reset. For example, writing a 1 to the `swwarmrstreq` bit in the `ctrl` register resets all the modules that have a checkmark in the Warm Reset column.

Note: For warm resets, software can set the warmmask registers to prevent the assertion of module reset signals to peripheral modules.

The column marked for Software Deassert denotes reset signals that are left asserted by the reset manager.

Table 3-4: MPU Group, Generated Module Resets

Module Reset Signal	Description	Reset Domain	Cold Reset	Warm Reset	Debug Reset	Software Deassert
`mpu_cpu_rst_n[0]`	Resets each processor in the MPU	System	X	X		
`mpu_cpu_rst_n[1]`	Resets each processor in the MPU	System	X	X		X

Module Reset Signal	Description	Reset Domain	Cold Reset	Warm Reset	Debug Reset	Software Deassert
mpu_wd_rst_n	Resets both per-processor watchdogs in the MPU	System	X	X		
mpu_scu_periph_rst_n	Resets Snoop Control Unit (SCU) and peripherals	System	X	X		

Table 3-5: PER1 Group, Generated Module Resets

Module Reset Signal	Description	Reset Domain	Cold Reset	Warm Reset	Debug Reset	Software Deassert
watchdog_rst_n[1:0]	Resets corresponding system watchdog timer	System	X	X		X
l4sys_timer_rst_n[1:0]	Resets corresponding OSC1 timer	System	X	X		X
sp_timer_rst_n[1:0]	Resets corresponding SP timer	System	X	X		X
i2c_rst_n[4:0]	Resets corresponding I^2C controller	System	X	X		X
uart_rst_n[1:0]	Resets corresponding UART	System	X	X		X
gpio_rst_n[2:0]	Resets corresponding GPIO interface	System	X	X		X
dma_periph_if_rst_n[7:0]	DMA controller request interface from FPGA fabric to DMA controller	System	X	X		X
emac_ptp_rst_n [1:0]	Resets corresponding EMAC precision time protocol	System	X	X		
emac_ecc_rst_n [2:0]	Resets corresponding to EMAC ECC	System	X	X		
usb_ecc_rst_n [1:0]	Resets corresponding to USB ECC	System	X	X		
nand_flash_ecc_rst_n	Resets corresponding to NAND Flash ECC	System	X	X		
qspi_flash_ecc_rst_n	Resets corresponding to QSPI Flash ECC	System	X	X		

Module Reset Signal	Description	Reset Domain	Cold Reset	Warm Reset	Debug Reset	Software Deassert
sdmmc_ecc_rst_n	Resets corresponding to SDMMC ECC	System	X	X		

Table 3-6: PER0 Group, Generated Module Resets

Module Reset Signal	Description	Reset Domain	Cold Reset	Warm Reset	Debug Reset	Software Deassert
emac_rst_n[2:0]	Resets corresponding EMAC	System	X	X		X
usb_rst_n[1:0]	Resets corresponding USB	System	X	X		X
nand_flash_rst_n	Resets NAND flash controller	System	X	X		X
qspi_flash_rst_n	Resets quad SPI flash controller	System	X	X		X
spim_rst_n[1:0]	Resets SPI master controller	System	X	X		X
spis_rst_n[1:0]	Resets SPI slave controller	System	X	X		X
sdmmc_rst_n	Resets SD/MMC controller	System	X	X		X
dma_rst_n	Resets DMA controller	System	X	X		X
dma_ecc_rst_n	Resets DMA ECC	System	X	X		

Table 3-7: Bridge Group, Generated Module Resets

Module Reset Signal	Description	Reset Domain	Cold Reset	Warm Reset	Debug Reset	Software Deassert
hps2fpga_bridge_rst_n	Resets HPS-to-FPGA AMBA® Advanced eXtensible Interface (AXI™) bridge	System	X	X		X
fpga2hps_bridge_rst_n	Resets FPGA-to-HPS AXI bridge	System	X	X		X
lwhps2fpga_bridge_rst_n	Resets lightweight HPS-to-FPGA AXI bridge	System	X	X		X
f2h_sdram_bridge0_rst_n	Resets SDRAM bridge 0	System	X	X		X

Module Reset Signal	Description	Reset Domain	Cold Reset	Warm Reset	Debug Reset	Software Deassert
f2h_sdram_bridge1_rst_n	Resets SDRAM bridge 1	System	X	X		X
f2h_sdram_bridge2_rst_n	Resets SDRAM bridge 2	System	X	X		X
ddr_scheduler_rst_n	Resets DDR scheduler	System	X	X		X

Table 3-8: MISC Group, Generated Module Resets

Group	Module Reset Signal	Description	Reset Domain	Cold Reset	Warm Reset	Debug Reset	Software Deassert
SYSMOD	boot_rom_rst_n	Resets boot ROM	System	X	X		
	onchip_ram_rst_nsy	Resets on-chip RAM	System	X	X		
	sys_manager_rst_n	Resets system manager (resets logic associated with cold or warm reset)	System	X			
	fpga_manager_rst_n	Resets FPGA manager	System	X	X		
	sys_dbg_rst_n	Resets debug masters and slaves connected to L3 interconnect and level 4 (L4) buses	System	X	X		
	onchip_ram_ecc_rst_n	Onchip RAM ECC reset	System	X	X		
	h2f_rst_n		System	X	X		
	sec_rst_n	Security reset	System	X			

Group	Module Reset Signal	Description	Reset Domain	Cold Reset	Warm Reset	Debug Reset	Software Deassert
COLD	sys_manager_cold_rst_n	Resets system manager (resets logic associated with cold reset only)	System	X			
	rst_pin_oe_rst	Pulls nRST pin low	System	X	X		
	timestamp_cold_rst_n	Resets debug timestamp to 0x0	System	X			
	clk_manager_cold_rst_n	Resets clock manager (resets logic associated with cold reset only)	System	X			
	tap_cold_rst_n	Resets portion of TAP controller in the DAP that must be reset on a cold reset	TAP	X			
	sec_cold_rst_n	Security cold reset	System	X			
	hmc_cold_rst_n	HMC cold reset	System	X	X		
	io_manager_cold_rst_n	I/O manager cold reset	System	X	X		
DBUG	dbg_rst_n	Resets debug components including DAP, trace, MPU debug logic, and any user debug logic in the FPGA fabric	Debug	X		X	
L3	l3_rst_n	Resets L3 Interconnect and L4 buses	System	X	X		

Table 3-9: RAM Clear Group, Generated Module Resets

Module Reset Signal	Reset Domain	Cold Reset	Warm Reset	Debug Reset	Software Deassert
onchip_sec_ram_rst_n	X	X	X		
otg0_sec_ram_rst_n		X	X		
otg1_sec_ram_rst_n		X	X		
sdmmc_sec_ram_rst_n		X	X		
emac0rx_sec_ram_rst_n		X	X		
emac0tx_sec_ram_rst_n		X	X		
emac1rx_sec_ram_rst_n		X	X		
emac1tx_sec_ram_rst_n		X	X		
emac2rx_sec_ram_rst_n		X	X		
emac2tx_sec_ram_rst_n		X	X		
dma_sec_ram_rst_n		X	X		
nandw_sec_ram_rst_n		X	X		
nandr_sec_ram_rst_n		X	X		
nande_sec_ram_rst_n		X	X		
qspi_sec_ram_rst_n		X	X		
mwp_sec_ram_rst_n		X	X		

Table 3-10: L3 Group, Generated Module Resets

Module Reset Signal	Description	Reset Domain	Cold Reset	Warm Reset	Debug Reset	Software Deassert
misc	Resets L3 interconnect and L4 buses	System	X	X		

Modules Requiring Software Deassert

The reset manager leaves the reset signal asserted on certain modules even after the rest of the HPS has come out of reset. These modules are likely to require software configuration before they can safely be taken out of reset.

When a module that has been held in reset is ready to start running, software can deassert the reset signal by writing to the appropriate register, shown in the following table.

Table 3-11: Module Reset Signals and Registers

HPS Peripheral	Reset Register	Module Reset Signal	Reset Group
MPU	mpumodrst	mpu_cpu_rst_n[1]	MPU
Watchdog	per1modrst	watchdog_rst_n[1:0]	PER1

HPS Peripheral	Reset Register	Module Reset Signal	Reset Group
Timer	per1modrst	l4sys_timer_rst_n[1:0]	PER1
Timer	per1modrst	sp_timer_rst_n[1:0]	PER1
I^2C	per1modrst	i2c_rst_n[4:0]	PER1
UART	per1modrst	uart_rst_n[1:0]	PER1
GPIO	per1modrst	gpio_rst_n[2:0]	PER1
DMA	per1modrst	dma_periph_if_rst_n[7:0]	PER1
Ethernet MAC	per0modrst	emac_rst_n[2:0]	PER0
USB 2.0 OTG	per0modrst	usb_rst_n[1:0]	PER0
NAND	per0modrst	nand_flash_rst_n	PER0
Quad SPI	per0modrst	qspi_flash_rst_n	PER0
SPI Master	per0modrst	spim_rst_n[1:0]	PER0
SPI Slave	per0modrst	spis_rst_n[1:0]	PER0
SD/MMC	per0modrst	sdmmc_rst_n	PER0
DMA	per0modrst	dma_rst_n	PER0
HPS-to-FPGA Bridge	brgmodrst	hps2fpga_bridge_rst_n	Bridge
FPGA-to-HPS Bridge	brgmodrst	fpga2hps_bridge_rst_n	Bridge
Lightweight HPS-to-FPGA Bridge	brgmodrst	lwhps2fpga_bridge_rst_n	Bridge
FPGA-to-SDRAM port 0	brgmodrst	f2s_sdram_bridge0_rst_n	Bridge
FPGA-to-SDRAM port 1	brgmodrst	f2s_sdram_bridge1_rst_n	Bridge
FPGA-to-SDRAM port 2	brgmodrst	f2s_sdram_bridge2_rst_n	Bridge
SDRAM Scheduler	brgmodrst	ddr_scheduler_rst_n	Bridge

Slave Interface and Status Register

The reset manager slave interface is used to control and monitor the reset states.

The status register (`stat`) in the reset manager contains the status of the reset requester. The register contains a bit for each monitored reset request. The `stat` register captures all reset requests that have occurred. Software is responsible for clearing the bits.

During the boot process, the Boot ROM copies the `stat` register value into memory before clearing it. After booting, you can read the value of the reset status register at memory address (`r0` + 0x0038). For more information, refer to the "HPS State on Entry to the Second-Stage Boot Loader" section of the *Booting and Configuration* appendix.

Related Information

HPS State on Entry to the Second-Stage Boot Loader on page 30-43

Functional Description of the Reset Manager

The reset manager generates reset signals to modules in the HPS and to the FPGA fabric. The following actions generate reset signals:

- Software writing a 1 to the `swcoldrstreq` or `swwarmrstreq` bits in the `ctrl` register. Writing either bit causes the reset controller to perform a reset sequence.
- Software writing to the `mpumodrst`, `per0modrst`, `per1modrst`, `brgmodrst`, `sysmodrst`, `coldmodrst`, `nrstmodrst` or `dbgmodrst` module reset control registers.
- Asserting reset request signals triggers the reset controller. All external reset requests cause the reset controller to perform a reset sequence.

Multiple reset requests can be driven to the reset manager at the same time. Cold reset requests take priority over warm and debug reset requests. Higher priority reset requests preempt lower priority reset requests. There is no priority difference among reset requests within the same domain.

If a cold reset request is issued while another cold reset is already underway, the reset manager extends the reset period for all the module reset outputs until all cold reset requests are removed. If a cold reset request is issued while the reset manager is removing other modules out of the reset state, the reset manager returns those modules back to the reset state.

If a warm reset request is issued while another warm reset is already underway, the first warm reset completes before the second warm reset begins. If the second warm reset request is removed before the first warm reset completes, the warm first reset is extended to meet the timing requirements of the second warm reset request.

The `nPOR` pin can be used to extend the cold reset beyond what the POR voltage monitor automatically provides. The use of the `nPOR` pin is optional and can be tied high when it is not required.

The nRST pin can be used to extend a warm reset. The nRST pin is tri-stated, and the Reset Manager stretches a warm reset by the count programed in the CSR (RSTMGR.COUNTS.NRSTCNT) [doc however you usually do registers and bits] After the count has elapsed, 256 additional clocks cycles elapse before the nRST input is sampled again.

The reset manager contains the stat register that indicates which reset source caused a reset as well as the `ramstat` register that indicates which RAM modules were cleared during the last reset. After a cold reset is complete, all bits are cleared except for the bit(s) that indicate the source of the cold reset. If multiple cold reset requests overlap with each other, the bit corresponding to the source that de-asserts its request last is set. If more than one source is de-asserted in the same cycle, the value of the bit corresponding to each source is set.

After a warm reset is complete, the bit(s) that indicate the source of the warm reset are set to 1. A warm reset doesn't clear any bits in the stat register, so bits corresponding to any reset source that has caused a warm reset since the last cold reset or since the bits were last cleared are set. Any bit can be manually cleared by writing a 1 to it.

If RAM memory is cleared during a Cold or Warm reset, then the `ramstat` register indicates which RAMs during the reset. Bits are cleared manually by writing a 1.

The hard memory controller (HMC) in the FPGA is reset by the reset manager using a GPIO. The HMC is reset only through software; resetting the HMC is not part of the operation of any reset sources.

Related Information

Arria 10 Device Datasheet

For information about the required duration of reset request signal assertion, refer to the Arria 10 Device Datasheet.

Reset Sequencing

The reset controller sequences resets without software assistance. Module reset signals are asserted asynchronously and synchronously. The reset manager deasserts the module reset signals synchronous to the `boot_clk` clock. Module reset signals are deasserted in groups in a fixed sequence. All module reset signals in a group are deasserted at the same time.

The reset manager sends a request to the clock manager to put the clocks in boot mode, which creates a fixed and known relationship between the `boot_clk` clock and all other clocks generated by the clock manager.

After the reset manager releases the MPU subsystem from reset, CPU1 is left in reset and CPU0 begins executing code from the reset vector address. Software is responsible for deasserting CPU1 and other resets, as shown in MPU Group, Generated Module Resets Table. Software deasserts resets by writing the `mpumodrst`, `per0modrst`, `per1modrst`, `brgmodrst`, `sysmodrst`, `coldmodrst`, `nrstmodrst`, and `dbgmodrst` module-reset control registers.

Software can also bypass the reset controller and generate reset signals directly through the module-reset control registers. In this case, software is responsible for asserting module reset signals, driving them for the appropriate duration, and deasserting them in the correct order. The clock manager is not typically in boot mode during this time, so software is responsible for knowing the relationship between the clocks generated by the clock manager. Software must not assert a module reset signal that would prevent software from deasserting the module reset signal. For example, software should not assert the module reset to the processor executing the software.

Table 3-12: Minimum Pulse Width

Reset Type	Value
Warm Reset	200 ns
Cold Reset	6 `osc1_clk` cycles

Figure 3-3: Cold Reset Timing Diagram

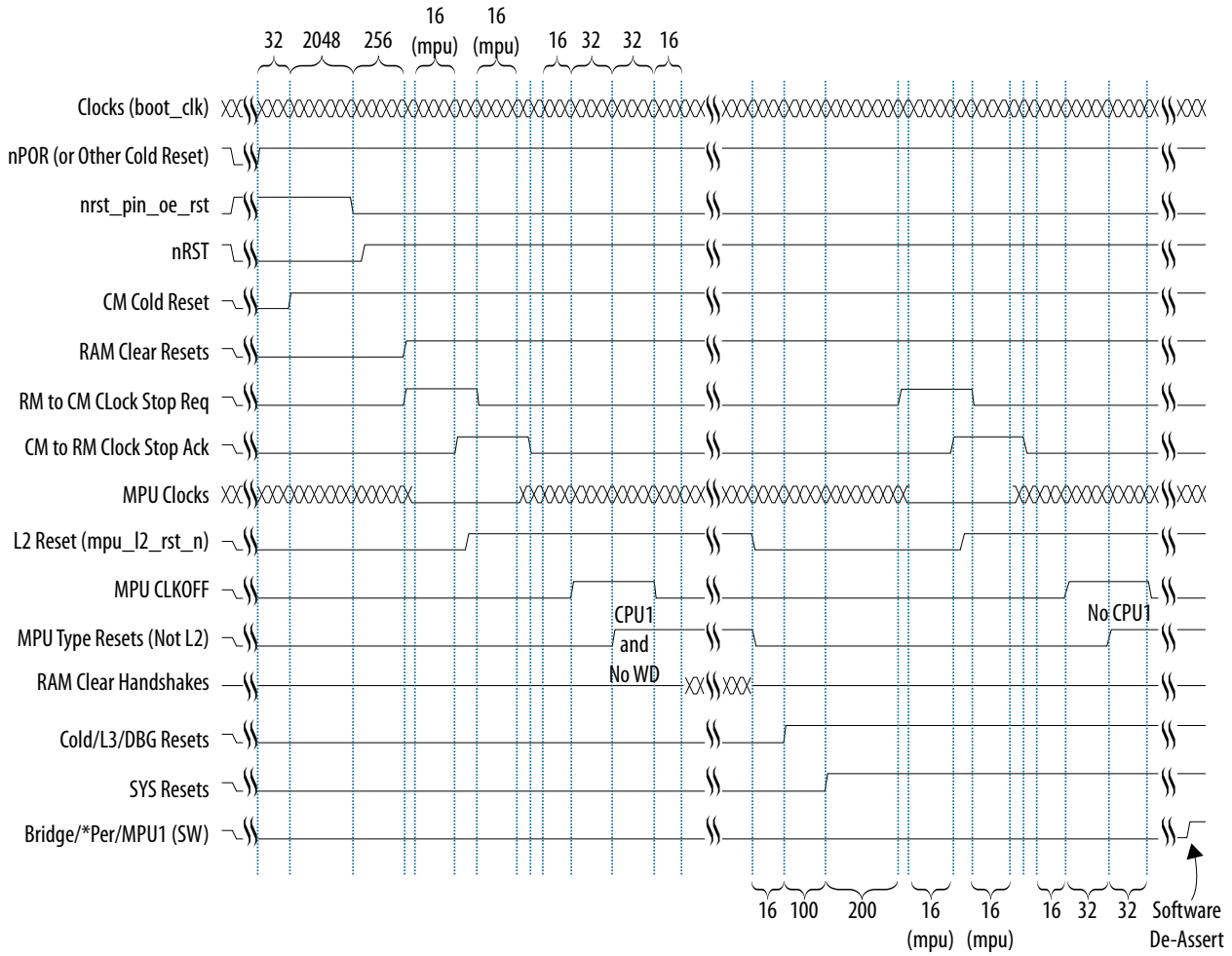
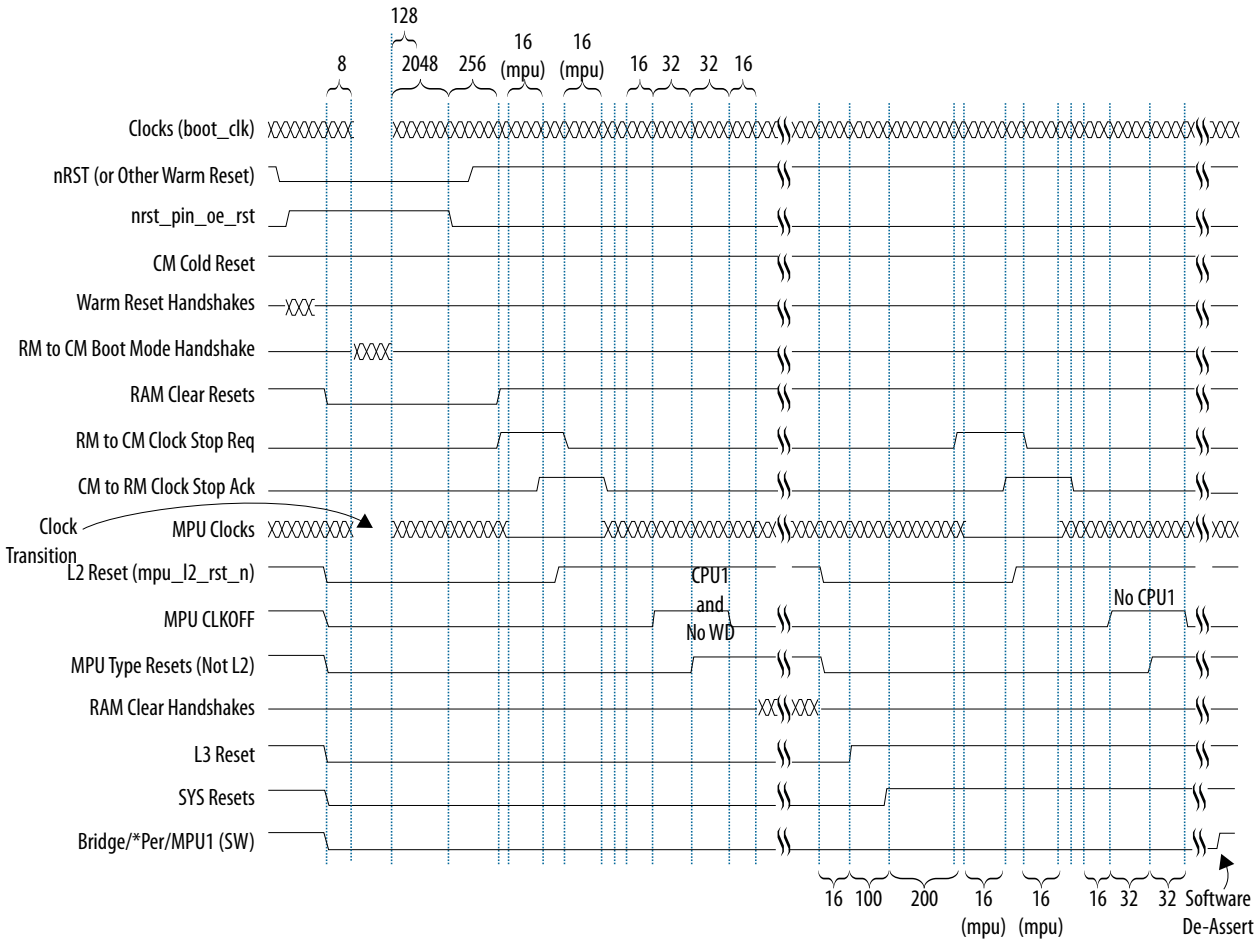


Figure 3-4: Warm Reset Timing Diagram



The cold and warm reset sequences consist of different reset assertion sequences and the same deassertion sequence. The following sections describe the sequences.

Note: Cold and warm reset affect only the `cpu0`, and by default cpu1 is held in reset until the software running in the `cpu0` releases it.

Related Information

- **Module Reset Signals** on page 3-8
- **Clock Manager** on page 2-1
  For more information about safe mode, refer to the *Clock Manager* chapter.

Cold Reset Assertion Sequence

The following list describes the assertion steps for cold reset shown in the Cold Reset timing diagram:

1. Assert all module resets.
2. Wait for level cold reset requests to de-assert
3. Wait for 32 cycles, de-asserts clock manager cold reset
4. Wait the nRST count (default is 2048). De-assert NRST Output Enable.

5. Wait 256 clocks to allow the nRST pin to stabilize. Start sampling nRST input pin.
6. Wait for level warm reset requests to all de-assert.
7. Go to de-assertion sequence.

Related Information

Cold and Warm Reset Deassertion Sequence on page 3-19

Warm Reset Assertion Sequence

The following list describes the assertion steps for warm reset shown in the Warm Reset Timing Diagram:

1. Reset manager performs optional handshake configured by SW with debug ETR. Wait for acknowledge.
2. Reset manager performs optional handshake configured by SW handshake with FPGA core. Wait for acknowledge.
3. Reset manager performs optional handshakes configured by SW with SDRAM and FPGA manager. Wait for acknowledges.
4. Reset manager asserts module resets except for MPU watchdogs if the watchdogs were the source of the warm reset.
5. Wait for 8 cycles, then reset manger sends `rm_cm_boot_mode_req` to clock manager. Wait until the Clock Manager acknowledges.
6. Start nRST count if non-zero and a fixed counter to 128 (COUNTS.WARMRSCYCLES).
7. If nRST count non-zero, start 256 stretch counter after nRST count is done to allow the nRST pin to stabilize. After 256 clocks, sample input pin nRST.
8. Wait for level warm reset requests to de-assert, the nRST count to be zero and the fixed 128 count to be zero.
9. Go to de-assertion sequence.

Related Information

Cold and Warm Reset Deassertion Sequence on page 3-19

Cold and Warm Reset Deassertion Sequence

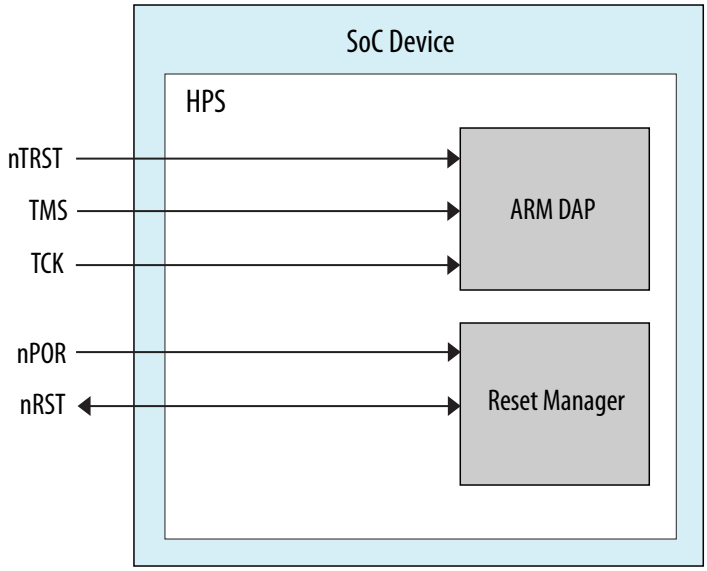
The following list describes the deassertion steps for both cold and warm reset shown in the Cold Reset Timing Diagram and Warm Reset Timing Diagram:

1. De-assert all RAM CLEAR resets. Reset Manager Initiates Clock Manager MPU clocks stop request.
2. Clock manager stops the MPU clocks and then waits 16 MPU clocks.
3. Clock manager asserts Clock stop acknowledge to Reset Manager.
4. Reset manager sees acknowledge and de-asserts L2 (mpu_l2_rst_n) reset only. Reset Manager de-asserts MPU stop request.
5. Clock manger sees request de-asserted. Wait 16 MPU clocks. Start MPU clocks, and Clock Manager de-asserts Clock stop acknowledge.
6. Reset manager sees de-assertion of acknowledge. Wait 16 cycles.
7. Reset manager asserts CPU0/1 CLKOFF signals and waits 32 cycles.
8. Reset manager de-asserts CPU0, CPU1, and SCU MPU resets (not WD). Wait 32 additional cycles.
9. De-assert CPU0/1 CLKOFF signals. Wait 16 cycles
10. Reset manager checks the Security fuse setting, and if the appropriate cold or warm reset fuse is set, do full RAM clearing sequence. Otherwise, clear only the MPU L1 RAM.
11. Wait for Security RAM Sequence to complete.
12. Assert again the CPU0, CPU1 and SCU MPU resets. Wait 16 cycles.
13. De-assert L3 reset, and if Cold Reset, de-assert DBG and COLD groups of resets. Wait for 100 cycles

14. De-assert SYS group module resets. Wait for 200 cycles.
15. Reset manager Initiates Clock Manager MPU clocks stop request.
16. Clock manager stops the MPU clocks and then waits 16 MPU clocks.
17. Clock manager asserts Clock stop acknowledge to Reset Manager.
18. Reset manager sees acknowledge and de-asserts L2 (mpu_l2_rst_n) reset only. Reset Manager de-asserts MPU stop request.
19. Clock manger sees request de-asserted. Wait 16 MPU clocks. Start MPU clocks, and Clock Manager de-asserts Clock stop acknowledge.
20. Reset manager sees de-assertion of acknowledge. Wait 16 cycles.
21. Reset manager asserts CPU0 CLKOFF signal(s) and waits 32 cycles.
22. Reset manager de-asserts remaining CPU0, SCU, and WD resets (no CPU1). Wait 32 additional cycles.
23. De-assert CPU0 CLKOFF signal(s). De-assertion sequence is finished.
24. Now SW should start running. SW to de-assert FPER/PER/SPER/BRIDGE/MPU2 resets.

Reset Pins

Figure 3-5: Reset Pins



The test reset (nTRST), test mode select (TMS), and test clock (TCK) pins are associated with the TAP reset domain and are used to reset the TAP controller in the DAP. These pins are not connected to the reset manager.

The nPOR and nRST pins are used to request cold and warm resets respectively. The nRST pin is an output as well. Any warm reset request causes the reset manager to drive the rst_pin_oe_rst output enable high, which drives the nRST pin low. The amount of time the reset manager pulls nRST low is controlled by the nRST pin count field (nrstcnt) of the reset cycles count register (counts); the default is 2048 clocks. This technique can be used to reset external devices (such as external memories) connected to the HPS.

Reset Effects

The following list describes how reset affects HPS logic:

- In the security manager domain, some status bits are cleared differently for cold and warm reset.
- The TAP reset domain ignores warm reset
- The Debug reset domain ignores warm reset
- The Clock Manager module is reset only by cold reset. After warm reset, the Clock Manager is put into Boot Mode.
- Each peripheral module and System Manager define reset behavior differently. See the appropriate control register for details.
- In the MPU module, the Watchdog Reset Status Registers are reset if an MPU Watchdog triggered the warm reset
- The Pin MUX is reset only by cold reset.

Reset Handshaking

The reset manager participates in several reset handshaking protocols to ensure other modules are safely reset.

Before issuing a warm reset, the reset manager performs a handshake with several modules to allow them to prepare for a warm reset. The handshake logic ensures the following conditions:

- Optionally the ETR master has no pending master transactions to the L3 interconnect
- Optionally preserve SDRAM contents during warm reset by issuing self-refresh mode request
- FPGA manager stops generating configuration clock
- Warns the FPGA fabric of the forthcoming warm reset

Similarly, the handshake logic associated with ETR also occurs during the debug reset to ensure that the ETR master has no pending master transactions to the L3 interconnect before the debug reset is issued. This action ensures that when ETR undergoes a debug reset, the reset has no adverse effects on the system domain portion of the ETR.

Reset Manager Address Map and Register Definitions

For complete HPS address map and register definitions, refer to the **Arria 10 HPS Address Map and Register Definitions**.

rst_mgr Address Map

Module Instance	Base Address	End Address
i_rst_mgr_rstmgr	0xFFD05000	0xFFD05FFF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
stat on page 3-30	0x0	32	RW	0x0	Status Register
ramstat on page 3-32	0x4	32	RW	0x0	RAM Status Register
miscstat on page 3-34	0x8	32	RW	0x0	Status Register
ctrl on page 3-36	0xC	32	RW	0x100000	Control Register
hdsken on page 3-37	0x10	32	RW	0x100000	Control Register
hdskreq on page 3-39	0x14	32	RW	0x100000	Control Register
hdskack on page 3-41	0x18	32	RW	0x100000	Control Register
counts on page 3-42	0x1C	32	RW	0x80000800	Reset Cycles Count Register
mpumodrst on page 3-43	0x20	32	RW	0x2	MPU Module Reset Register
per0modrst on page 3-45	0x24	32	RW	0xFF7FFFFF	Peripheral 0 Module Reset Register
per1modrst on page 3-48	0x28	32	RW	0x7031F3F	Peripheral Module Reset Register
brgmodrst on page 3-50	0x2C	32	RW	0x7F	Bridge Module Reset Register
sysmodrst on page 3-51	0x30	32	RW	0x0	SYSTEM Module Reset Register
coldmodrst on page 3-52	0x34	32	RW	0x0	COLD Module Reset Register
nrstmodrst on page 3-54	0x38	32	RW	0x0	NRST Module Reset Register
dbgmodrst on page 3-55	0x3C	32	RW	0x0	Debug Module Reset Register
mpuwarmmask on page 3-56	0x40	32	RW	0x1F	MPU Warm Mask Register
per0warmmask on page 3-57	0x44	32	RW	0xFF7FFFFF	Peripheral 0 Warm Mask Register

Register	Offset	Width	Access	Reset Value	Description
per1warmmask on page 3-60	0x48	32	RW	0x7031F3F	Peripheral 1 Warm Mask Register
brgwarmmask on page 3-62	0x4C	32	RW	0x7F	Bridge Warm Mask Register
syswarmmask on page 3-63	0x50	32	RW	0x1FF	SYSTEM Warm Mask Register
nrstwarmmask on page 3-65	0x54	32	RW	0x1	NRST Warm Mask Register
l3warmmask on page 3-66	0x58	32	RW	0x1	Mask L3 Register
tststa on page 3-67	0x5C	32	RO	0x0	Test Status
tstscratch on page 3-67	0x60	32	RW	0x0	Test Scratch
hdsktimeout on page 3-68	0x64	32	RW	0x2800	Hand Shake Time Out
hmcintr on page 3-69	0x68	32	RW	0x0	HMC Interrupt
hmcintren on page 3-69	0x6C	32	RW	0x0	HMC Interrupt enable
hmcintrens on page 3-70	0x70	32	RW	0x0	HMC Interrupt enable set
hmcintrenr on page 3-71	0x74	32	RW	0x0	HMC Interrupt Enable Clear
hmcgpout on page 3-71	0x78	32	RW	0x0	HMC GPIO Output
hmcgpin on page 3-72	0x7C	32	RO	0x0	HMC GPIO Input

rst_mgr Summary

Base Address: 0xFFD05000

Register Address Offset	Bit Fields
i_rst_mgr_rstmgr	

Register Address Offset	Bit Fields															
stat 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved														cdbgreqrst RW 0x0	fpgadbgrst RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	l4wd1rst RW 0x0	l4wd0rst RW 0x0	mpuwd1rst RW 0x0	mpuwd0rst RW 0x0	swwarmrst RW 0x0	fpgawarmrst RW 0x0	nrstpinrst RW 0x0	Reserved		swcoldrst RW 0x0	configiocoldrst RW 0x0	fpgacoldrst RW 0x0	nporpinrst RW 0x0	porfpgavoltrst RW 0x0	porhpsvoltrst RW 0x0
ramstat 0x4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved														mpul1timeout RW 0x0	mpul1ramclr RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	mwpramclr RW 0x0	qspiramclr RW 0x0	emac2rxramclr RW 0x0	emac2txramclr RW 0x0	emac1txramclr RW 0x0	emac1rxramclr RW 0x0	emac0txramclr RW 0x0	emac0rxramclr RW 0x0	nanderamclr RW 0x0	nandrramclr RW 0x0	nandwramclr RW 0x0	dmaramclr RW 0x0	sdmmcramclr RW 0x0	otg1ramclr RW 0x0	otg0ramclr RW 0x0	onchipramclr RW 0x0
miscstat 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved												etrstalltimeout RW 0x0	fpgahstimeout RW 0x0	fpgamgrhstimeout RW 0x0	sdrselfreftimeout RW 0x0
ctrl 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														swwarmrstreq RW 0x0	swcoldr-streq RW 0x0

Register Address Offset	Bit Fields															
hdsken 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved												etrstallen RW 0x0	fpgahsen RW 0x0	fpgamgrhsen RW 0x0	sdrselfrefen RW 0x0
hdskreq 0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved												etrstallreq RW 0x0	fpgahsreq RW 0x0	fpgamgrhsreq RW 0x0	sdrselfrefreq RW 0x0
hdskack 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							etrstallwarmrst RW 0x0	Reserved				etrstallack RO 0x0	fpgahsack RO 0x0	fpgamgrhsack RO 0x0	sdrselfreqack RO 0x0
counts 0x1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	warmrstcycles RW 0x80								Reserved				nrstcnt RW 0x800			
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	nrstcnt RW 0x800															
mpumodrst 0x20	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved												scuper RW 0x0	wds RW 0x0	cpu1 RW 0x1	cpu0 RW 0x0

Register Address Offset	Bit Fields															
per0modrst 0x24	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	dmaif7 RW 0x1	dmaif6 RW 0x1	dmaif5 RW 0x1	dmaif4 RW 0x1	dmaif3 RW 0x1	dmaif2 RW 0x1	dmaif1 RW 0x1	dmaif0 RW 0x1	Reserved	emacptp RW 0x1	dmaocp RW 0x1	spis1 RW 0x1	spis0 RW 0x1	spim1 RW 0x1	spim0 RW 0x1	dma RW 0x1
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	sdmmcocp RW 0x1	qspiocp RW 0x1	nandocp RW 0x1	usb1ocp RW 0x1	usb0ocp RW 0x1	emac2ocp RW 0x1	emac1ocp RW 0x1	emac0ocp RW 0x1	sdmmc RW 0x1	qspi RW 0x1	nand RW 0x1	usb1 RW 0x1	usb0 RW 0x1	emac2 RW 0x1	emac1 RW 0x1	emac0 RW 0x1
per1modrst 0x28	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved					gpio2 RW 0x1	gpio1 RW 0x1	gpio0 RW 0x1	Reserved						uart1 RW 0x1	uart0 RW 0x1
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved			i2c4 RW 0x1	i2c3 RW 0x1	i2c2 RW 0x1	i2c1 RW 0x1	i2c0 RW 0x1	Reserved		sptimer1 RW 0x1	sptimer0 RW 0x1	l4systimer1 RW 0x1	l4systimer0 RW 0x1	watchdog1 RW 0x1	watchdog0 RW 0x1
brgmodrst 0x2C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved									ddrsch RW 0x1	f2ssdram2 RW 0x1	f2ssdram1 RW 0x1	f2ssdram0 RW 0x1	fpga2hps RW 0x1	lwhps2fpga RW 0x1	hps2fpga RW 0x1
sysmodrst 0x30	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved									ocramocp RW 0x0	sysdbg RW 0x0	s2f RW 0x0	fpgamgr RW 0x0	Reserved	ocram RW 0x0	rom RW 0x0

Register Address Offset	Bit Fields															
coldmodrst 0x34	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								iomgrcold RW 0x0	hmccold RW 0x0	tapcold RW 0x0	timestampcold RW 0x0	s2fcold RW 0x0	Reserved		clkmgrcold RW 0x0
nrstmodrst 0x38	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															nrstpinoe RW 0x0
dbgmodrst 0x3C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															dbg RW 0x0
mpuwarmmask 0x40	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															wds RW 0x1
per0warmmask 0x44	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	dmaif7 RW 0x1	dmaif6 RW 0x1	dmaif5 RW 0x1	dmaif4 RW 0x1	dmaif3 RW 0x1	dmaif2 RW 0x1	dmaif1 RW 0x1	dmaif0 RW 0x1	Reserved	emacptp RW 0x1	dmaocp RW 0x1	spis1 RW 0x1	spis0 RW 0x1	spim1 RW 0x1	spim0 RW 0x1	dma RW 0x1
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	sdmmcocp RW 0x1	qspiocp RW 0x1	nandocp RW 0x1	usb1ocp RW 0x1	usb0ocp RW 0x1	emac2ocp RW 0x1	emac1ocp RW 0x1	emac0ocp RW 0x1	sdmmc RW 0x1	qspi RW 0x1	nand RW 0x1	usb1 RW 0x1	usb0 RW 0x1	emac2 RW 0x1	emac1 RW 0x1	emac0 RW 0x1

Register Address Offset	Bit Fields															
per1warmmask 0x48	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved					gpio2 RW 0x1	gpio1 RW 0x1	gpio0 RW 0x1	Reserved						uart1 RW 0x1	uart0 RW 0x1
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved			i2c4 RW 0x1	i2c3 RW 0x1	i2c2 RW 0x1	i2c1 RW 0x1	i2c0 RW 0x1	Reserved		sptimer1 RW 0x1	sptimer0 RW 0x1	l4systimer1 RW 0x1	l4systimer0 RW 0x1	watchdog1 RW 0x1	watchdog0 RW 0x1
brgwarmmask 0x4C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved									ddrsch RW 0x1	f2ssdram2 RW 0x1	f2ssdram1 RW 0x1	f2ssdram0 RW 0x1	fpga2hps RW 0x1	lwhps2fpga RW 0x1	hps2fpga RW 0x1
syswarmmask 0x50	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved									ocramocp RW 0x1	sysdbg RW 0x1	s2f RW 0x1	fpgamgr RW 0x1	Reserved	ocram RW 0x1	rom RW 0x1
nrstwarm-mask 0x54	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															nrstpinoe RW 0x1
l3warmmask 0x58	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															l3 RW 0x1

Register Address Offset	Bit Fields															
tststa 0x5C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved									dbgrstst RO 0x0			warmrstst RO 0x0			
tstscratch 0x60	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	fld0 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	fld0 RW 0x0															
hdsktimeout 0x64	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved							val RW 0x2800								
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	val RW 0x2800															
hmcintr 0x68	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															intr RW 0x0
hmcintren 0x6C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															en RW 0x0
hmcintrens 0x70	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															en RW 0x0

Register Address Offset	Bit Fields															
hmcintrenr 0x74	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															en RW 0x0
hmcgpout 0x78	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								out RW 0x0							
hmcgpin 0x7C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								in RO 0x0							

stat

The STAT register contains bits that indicate the reset source. For reset sources, a field is 1 if its associated reset requester caused the reset.

Software clears bits by writing them with a value of 1. Writes to bits with a value of 0 are ignored.

After a cold reset is complete, all bits are reset to their reset value except for the bit(s) that indicate the source of the cold reset. If multiple cold reset requests overlap with each other, the source de-asserts the request last will be logged. The other reset request source(s) de-assert the request in the same cycle will also be logged, the rest of the fields are reset to default value of 0.

After a warm reset is complete, the bit(s) that indicate the source of the warm reset are set to 1. A warm reset doesn't clear any of the bits in the STAT register; these bits must be cleared by software writing the STAT register.

Module Instance	Base Address	Register Address
i_rst_mgr_rstmgr	0xFFD05000	0xFFD05000

Offset: 0x0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved														cdbgreqrst RW 0x0	fpgadbgrst RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	l4wd1rst RW 0x0	l4wd0rst RW 0x0	mpuwd1rst RW 0x0	mpuwd0rst RW 0x0	swwarmrst RW 0x0	fpgawarmrst RW 0x0	nrstpinrst RW 0x0	Reserved		swcoldrst RW 0x0	configioocldrst RW 0x0	fpgacoldrst RW 0x0	nporpinrst RW 0x0	porfpgavoltrst RW 0x0	porhpsvoltrst RW 0x0

stat Fields

Bit	Name	Description	Access	Reset
17	cdbgreqrst	DAP triggered debug reset	RW	0x0
16	fpgadbgrst	FPGA triggered debug reset	RW	0x0
14	l4wd1rst	L4 Watchdog 1 triggered a hardware sequenced warm reset	RW	0x0
13	l4wd0rst	L4 Watchdog 0 triggered a hardware sequenced warm reset	RW	0x0
12	mpuwd1rst	MPU Watchdog 1 triggered a hardware sequenced warm reset	RW	0x0
11	mpuwd0rst	MPU Watchdog 0 triggered a hardware sequenced warm reset	RW	0x0
10	swwarmrst	Software wrote CTRL.SWARMRSTREQ to 1 and triggered a hardware sequenced warm reset.	RW	0x0
9	fpgawarmrst	FPGA core triggered a hardware sequenced warm reset	RW	0x0
8	nrstpinrst	nRST pin triggered a hardware sequenced warm reset	RW	0x0

Bit	Name	Description	Access	Reset
5	swcoldrst	Software wrote CTRL.SWCOLDRSTREQ to 1 and triggered a cold reset.	RW	0x0
4	configiocoldrst	FPGA entered CONFIG_IO mode and a triggered a cold reset	RW	0x0
3	fpgacoldrst	FPGA core triggered a cold reset (f2s_cold_rst_req = 1)	RW	0x0
2	nporpinrst	nPOR pin triggered a cold reset (por_pin_req = 1)	RW	0x0
1	porfpgavoltrst	Built-in FPGA POR voltage detector triggered a cold reset. Security Manager brought Reset Manager out of POR Reset.	RW	0x0
0	porhpsvoltrst	Built-in HPS POR voltage detector triggered a cold reset. Security Manager brought Reset Manager out of POR Reset.	RW	0x0

ramstat

The RAMSTAT register contains bits that indicate the security RAM clearing event during cold or warm reset for each RAM.

Software clears bits by writing them with a value of 1. Writes to bits with a value of 0 are ignored.

For MPU, there are seperate bits for L1 invalidate only or full security clearing.

There is another bit for L1 invalidate timeout error only. The security RAM clearing does not have a timeout.

Module Instance	Base Address	Register Address
i_rst_mgr_rstmgr	0xFFD05000	0xFFD05004

Offset: 0x4

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved														mpul1timeout RW 0x0	mpul1ramclr RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
mwpramclr RW 0x0	qspiramclr RW 0x0	emac2rxramclr RW 0x0	emac2txramclr RW 0x0	emac1txramclr RW 0x0	emac1rxramclr RW 0x0	emac0txramclr RW 0x0	emac0rxramclr RW 0x0	nanderamclr RW 0x0	nandrramclr RW 0x0	nandwramclr RW 0x0	dmaramclr RW 0x0	sdmmcramclr RW 0x0	otg1ramclr RW 0x0	otg0ramclr RW 0x0	onchipramclr RW 0x0

ramstat Fields

Bit	Name	Description	Access	Reset
17	mpul1timeout	RAMSTATUS bit to indicate MPU l1 RAM cleared timeout during cold/warm reset	RW	0x0
16	mpul1ramclr	RAMSTATUS bit to indicate MPU L1 invalidate clearing only or full security clearing during cold/warm reset	RW	0x0
15	mwpramclr	RAMSTATUS bit to indicate MWP RAM is cleared during cold/warm reset	RW	0x0
14	qspiramclr	RAMSTATUS bit to indicate QSPI RAM is cleared during cold/warm reset	RW	0x0
13	emac2rxramclr	RAMSTATUS bit to indicate EMAC2 RX RAM is cleared during cold/warm reset	RW	0x0
12	emac2txramclr	RAMSTATUS bit to indicate EMAC2 TX RAM is cleared during cold/warm reset	RW	0x0
11	emac1txramclr	RAMSTATUS bit to indicate EMAC1 TX RAM is cleared during cold/warm reset	RW	0x0
10	emac1rxramclr	RAMSTATUS bit to indicate EMAC1 RX RAM is cleared during cold/warm reset	RW	0x0

Bit	Name	Description	Access	Reset
9	emac0txramclr	RAMSTATUS bit to indicate EMAC0 TX RAM is cleared during cold/warm reset	RW	0x0
8	emac0rxramclr	RAMSTATUS bit to indicate EMAC0 RX RAM is cleared during cold/warm reset	RW	0x0
7	nanderamclr	RAMSTATUS bit to indicate NAND ECC RAM is cleared during cold/warm reset	RW	0x0
6	nandrramclr	RAMSTATUS bit to indicate NAND Read RAM is cleared during cold/warm reset	RW	0x0
5	nandwramclr	RAMSTATUS bit to indicate NAND Write RAM is cleared during cold/warm reset	RW	0x0
4	dmaramclr	RAMSTATUS bit to indicate DMA RAM is cleared during cold/warm reset	RW	0x0
3	sdmmcramclr	RAMSTATUS bit to indicate SDMMC RAM is cleared during cold/warm reset	RW	0x0
2	otg1ramclr	RAMSTATUS bit to indicate USB1 RAM is cleared during cold/warm reset	RW	0x0
1	otg0ramclr	RAMSTATUS bit to indicate USB0 RAM is cleared during cold/warm reset	RW	0x0
0	onchipramclr	RAMSTATUS bit to indicate Onchip RAM is cleared during cold/warm reset	RW	0x0

miscstat

The MISCSTAT register contains bits that indicate the timeout event. For timeout events, a field is 1 if its associated timeout occured as part of a hardware sequenced warm/debug reset.

Software clears bits by writing them with a value of 1. Writes to bits with a value of 0 are ignored.

After a cold reset is complete, all bits are reset to their reset value except for the bit(s) that indicate the source of the cold reset. If multiple cold reset requests overlap with each other, the source de-asserts the request last will be logged. The other reset

request source(s) de-assert the request in the same cycle will also be logged, the rest of the fields are reset to default value of 0.

After a warm reset is complete, the bit(s) that indicate the source of the warm reset are set to 1. A warm reset doesn't clear any of the bits in the MISCSTAT register; these bits must be cleared by software writing the STAT register.

Module Instance	Base Address	Register Address
i_rst_mgr_rstmgr	0xFFD05000	0xFFD05008

Offset: 0x8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												etrstalltimeout RW 0x0	fpgahstimeout RW 0x0	fpgamgrhstimeout RW 0x0	sdrselfreftimeout RW 0x0

miscstat Fields

Bit	Name	Description	Access	Reset
3	etrstalltimeout	A 1 indicates that Reset Manager's request to the ETR (Embedded Trace Router) to stall its AXI master port before starting a hardware sequenced warm reset timed-out and the Reset Manager had to proceed with the warm reset anyway.	RW	0x0
2	fpgahstimeout	A 1 indicates that Reset Manager's handshake request to FPGA before starting a hardware sequenced warm reset timed-out and the Reset Manager had to proceed with the warm reset anyway.	RW	0x0

Bit	Name	Description	Access	Reset
1	fpgamgrhstimeout	A 1 indicates that Reset Manager's request to the FPGA manager to stop driving configuration clock to FPGA CB before starting a hardware sequenced warm reset timed-out and the Reset Manager had to proceed with the warm reset anyway.	RW	0x0
0	sdrselfreftimeout	A 1 indicates that Reset Manager's request to the SDRAM Controller Subsystem to put the SDRAM devices into self-refresh mode before starting a hardware sequenced warm reset timed-out and the Reset Manager had to proceed with the warm reset anyway.	RW	0x0

ctrl

The CTRL register is used by software to control reset behavior.It includes fields for software to initiate the cold and warm reset.

Module Instance	Base Address	Register Address
i_rst_mgr_rstmgr	0xFFD05000	0xFFD0500C

Offset: 0xC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														swwarmrstreq RW 0x0	swcoldrstreq RW 0x0

ctrl Fields

Bit	Name	Description	Access	Reset
1	swwarmrstreq	This is a one-shot bit written by software to 1 to trigger a hardware sequenced warm reset. It always reads the value 0.	RW	0x0
0	swcoldrstreq	This is a one-shot bit written by software to 1 to trigger a cold reset. It always reads the value 0.	RW	0x0

hdsken

The CTRL register is used by software to control reset behavior.It includes fields for enable hardware handshake with other modules before warm reset.

Module Instance	Base Address	Register Address
i_rst_mgr_rstmgr	0xFFD05000	0xFFD05010

Offset: 0x10

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												etrstallen RW 0x0	fpgahsen RW 0x0	fpgamgrhsen RW 0x0	sdrselfrefen RW 0x0

hdsken Fields

Bit	Name	Description	Access	Reset
3	etrstallen	Software writes this field 1 to request to the ETR that it stalls its AXI master to the L3 Interconnect. Software waits for the ETRSTALLACK to be 1 and then writes this field to 0. Note that it is possible for the ETR to never assert ETRSTALLACK so software should timeout if ETRSTALLACK is never asserted.	RW	0x0
2	fpgahsen	This field controls whether to perform handshake with FPGA before asserting warm reset. If set to 1, the Reset Manager makes a request to the FPGAbefore asserting warm reset signals. However if FPGA is already in warm reset state, the handshake is not performed. If set to 0, the handshake is not performed	RW	0x0
1	fpgamgrhsen	Enables a handshake between the Reset Manager and FPGA Manager before a warm reset. The handshake is used to warn the FPGA Manager that a warm reset it coming so it can prepare for it. When the FPGA Manager receives a warm reset handshake, the FPGA Manager drives its output clock to a quiescent state to avoid glitches. If set to 1, the Manager makes a request to the FPGA Managerbefore asserting warm reset signals. However if the FPGA Manager is already in warm reset, the handshake is skipped. If set to 0, the handshake is skipped.	RW	0x0

Bit	Name	Description	Access	Reset
0	sdrselfrefen	This field controls whether the contents of SDRAM devices survive a hardware sequenced warm reset. If set to 1, the Reset Manager makes a request to the SDRAM Controller Subsystem to put the SDRAM devices into self-refresh mode before asserting warm reset signals. However, if SDRAM is already in warm reset, Handshake with SDRAM is not performed.	RW	0x0

hdskreq

The CTRL register is used by software to control reset behavior.It includes fields for software to initiate the cold and warm reset, enable hardware handshake with other modules before warm reset, and perform software handshake. The software handshake sequence must match the hardware sequence. Software mustde-assert the handshake request after asserting warm reset and before de-assert the warm reset.

Fields are only reset by a cold reset.

Module Instance	Base Address	Register Address
i_rst_mgr_rstmgr	0xFFD05000	0xFFD05014

Offset: 0x14

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												etrstallreq RW 0x0	fpgahsreq RW 0x0	fpgamgrhsreq RW 0x0	sdrselfrefreq RW 0x0

hdskreq Fields

Bit	Name	Description	Access	Reset
3	etrstallreq	Software writes this field 1 to request to the ETR that it stalls its AXI master to the L3 Interconnect. Software waits for the ETRSTALLACK to be 1 and then writes this field to 0. Note that it is possible for the ETR to never assert ETRSTALLACK so software should timeout if ETRSTALLACK is never asserted.	RW	0x0
2	fpgahsreq	Software writes this field 1 to initiate handshake request to FPGA . Software waits for the FPGAHSACK to be active and then writes this field to 0. Note that it is possible for the FPGA to never assert FPGAHSACK so software should timeout in this case.	RW	0x0
1	fpgamgrhsreq	Software writes this field 1 to request to the FPGA Manager to idle its output clock. Software waits for the FPGAMGRHSACK to be 1 and then writes this field to 0. Note that it is possible for the FPGA Manager to never assert FPGAMGRHSACK so software should timeout in this case.	RW	0x0
0	sdrselfrefreq	Software writes this field 1 to request to the SDRAM Controller Subsystem that it puts the SDRAM devices into self-refresh mode. This is done to preserve SDRAM contents across a software warm reset. Software waits for the SDRSELFREFACK to be 1 and then writes this field to 0. Note that it is possible for the SDRAM Controller Subsystem to never assert SDRSELFREFACK so software should timeout if SDRSELFREFACK is never asserted.	RW	0x0

hdskack

The CTRL register is used by software to control reset behavior.It includes fields for software to initiate the cold and warm reset, enable hardware handshake with other modules before warm reset, and perform software handshake. The software handshake sequence must match the hardware sequence. Software mustde-assert the handshake request after asserting warm reset and before de-assert the warm reset.

Fields are only reset by a cold reset.

Module Instance	Base Address	Register Address
i_rst_mgr_rstmgr	0xFFD05000	0xFFD05018

Offset: 0x18

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							etrstallwarmrst RW 0x0	Reserved				etrstallack RO 0x0	fpgahsack RO 0x0	fpgamgrhsack RO 0x0	sdrselfreqack RO 0x0

hdskack Fields

Bit	Name	Description	Access	Reset
8	etrstallwarmrst	If a warm reset occurs and ETRSTALLEN is 1, hardware sets this bit to 1 to indicate that the stall of the ETR AXI master is pending. Hardware leaves the ETR stalled until software clears this field by writing it with 1. Software must only clear this field when it is ready to have the ETR AXI master start making AXI requests to write trace data.	RW	0x0
3	etrstallack	This is the acknowlege for a ETR AXI master stall initiated by the ETRSTALLREQ field. A 1 indicates that the ETR has stalled its AXI master	RO	0x0
2	fpgahsack	This is the acknowlege (high active) that the FPGA handshake acknowledge has been received by Reset Manager.	RO	0x0
1	fpgamgrhsack	This is the acknowlege (high active) that the FPGA manager has successfully idled its output clock.	RO	0x0
0	sdrselfreqack	This is the acknowlege for a SDRAM self-refresh mode request initiated by the SDRSELFREFREQ field. A 1 indicates that the SDRAM Controller Subsystem has put the SDRAM devices into self-refresh mode.	RO	0x0

counts

The COUNTS register is used by software to control reset behavior.It includes fields for software to control the behavior of the warm reset and nRST pin.

Fields are only reset by a cold reset.

Module Instance	Base Address	Register Address
i_rst_mgr_rstmgr	0xFFD05000	0xFFD0501C

Offset: 0x1C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
warmrstcycles RW 0x80								Reserved				nrstcnt RW 0x800			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
nrstcnt RW 0x800															

counts Fields

Bit	Name	Description	Access	Reset
31:24	warmrstcycles	On a warm reset, the Reset Manager releases the reset to the Clock Manager, and then waits for the number of cycles specified in this register before releasing the rest of the hardware controlled resets.	RW	0x80
19:0	nrstcnt	The Reset Manager pulls down the nRST pin on a warm reset for the number of cycles specified in this register. A value of 0x0 is reserved. nRST assertion on a warm reset may be disabled by clearing NRSTWARMMASK.NRSTPINOE.	RW	0x800

mpumodrst

The MPUMODRST register is used by software to trigger module resets (individual module reset signals). Software explicitly asserts and de-asserts module reset signals by writing bits in the appropriate *MODRST register. It is up to software to ensure module reset signals are asserted for the appropriate length of time and are de-asserted in the correct order. It is also up to software to not assert a module reset signal that would prevent software from de-asserting the module reset signal. For example, software should not assert the module reset to the CPU executing the software.

Software writes a bit to 1 to assert the module reset signal and to 0 to de-assert the module reset signal.

All fields except CPU1 are only reset by a cold reset. The CPU1 field is reset by a cold reset. The CPU1 field is also reset by a warm reset if not masked by the corresponding MPUWARMMASK field.

Module Instance	Base Address	Register Address
i_rst_mgr_rstmgr	0xFFD05000	0xFFD05020

Offset: 0x20

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												scuper RW 0x0	wds RW 0x0	cpu1 RW 0x1	cpu0 RW 0x0

mpumodrst Fields

Bit	Name	Description	Access	Reset
3	scuper	Resets SCU and peripherals. Peripherals consist of the interrupt controller, global timer, both per-CPU private timers, and both per-CPU watchdogs (except for the Watchdog Reset Status registers).	RW	0x0
2	wds	Resets both per-CPU Watchdog Reset Status registers in MPU.	RW	0x0

Bit	Name	Description	Access	Reset
1	cpu1	Resets Cortex-A9 CPU1 in MPU. It is reset to 1 on a cold or warm reset. This holds CPU1 in reset until software is ready to release CPU1 from reset by writing 0 to this field. On single-core devices, writes to this field are ignored.On dual-core devices, writes to this field trigger the same sequence as writes to the CPU0 field (except the sequence is performed on CPU1).	RW	0x1
0	cpu0	Resets Cortex-A9 CPU0 in MPU. Whe software changes this field from 0 to 1, ittriggers the following sequence: 1. CPU0 reset is asserted. cpu0 clkoff is de-asserted 2. after 32 osc1_clk cycles, cpu0 clkoff is asserted. When software changes this field from 1 to 0, it triggers the following sequence: 1.CPU0 reset is de-asserted. 2. after 32 cycles, cpu0 clkoff is de-asserted. Software needs to wait for at least 64 osc1_clk cycles between each change of this field to keep the proper reset/clkoff sequence.	RW	0x0

per0modrst

The PER0MODRST register is used by software to trigger module resets for Peripheral Group and Fast Peripheral Group. Regular Peripheral Group resets are in the lower 2 bytes, and Fast Pheripheral Group resets are in the upper 2 bytes. Software explicitly asserts and de-asserts module reset signals by writing bits in the appropriate *MODRST register. It is up to software to ensure module reset signals are asserted for the appropriate length of time and are de-asserted in the correct order. It is also up to software to not assert a module reset signal that would prevent software from de-asserting the module reset signal. For example, software should not assert the module reset to the CPU executing the software.

Software writes a bit to 1 to assert the module reset signal and to 0 to de-assert the module reset signal.

All fields are reset by a cold reset.All fields are also reset by a warm reset if not masked by the corresponding PERWARMMASK field. The reset value of all fields is 1. This holds the corresponding

module in reset until software is ready to release the module from reset by writing 0 to its field.

Module Instance	Base Address	Register Address
i_rst_mgr_rstmgr	0xFFD05000	0xFFD05024

Offset: 0x24

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dmaif7 RW 0x1	dmaif6 RW 0x1	dmaif5 RW 0x1	dmaif4 RW 0x1	dmaif3 RW 0x1	dmaif2 RW 0x1	dmaif1 RW 0x1	dmaif0 RW 0x1	Reserved	emacptp RW 0x1	dmaocp RW 0x1	spis1 RW 0x1	spis0 RW 0x1	spim1 RW 0x1	spim0 RW 0x1	dma RW 0x1
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
sdmmcocp RW 0x1	qspiocp RW 0x1	nandocp RW 0x1	usb1ocp RW 0x1	usb0ocp RW 0x1	emac2ocp RW 0x1	emac1ocp RW 0x1	emac0ocp RW 0x1	sdmmc RW 0x1	qspi RW 0x1	nand RW 0x1	usb1 RW 0x1	usb0 RW 0x1	emac2 RW 0x1	emac1 RW 0x1	emac0 RW 0x1

per0modrst Fields

Bit	Name	Description	Access	Reset
31	dmaif7	Resets DMA channel 7 interface adapter between FPGA Fabric and HPS DMA Controller	RW	0x1
30	dmaif6	Resets DMA channel 6 interface adapter between FPGA Fabric and HPS DMA Controller	RW	0x1
29	dmaif5	Resets DMA channel 5 interface adapter between FPGA Fabric and HPS DMA Controller	RW	0x1
28	dmaif4	Resets DMA channel 4 interface adapter between FPGA Fabric and HPS DMA Controller	RW	0x1

Bit	Name	Description	Access	Reset
27	dmaif3	Resets DMA channel 3 interface adapter between FPGA Fabric and HPS DMA Controller	RW	0x1
26	dmaif2	Resets DMA channel 2 interface adapter between FPGA Fabric and HPS DMA Controller	RW	0x1
25	dmaif1	Resets DMA channel 1 interface adapter between FPGA Fabric and HPS DMA Controller	RW	0x1
24	dmaif0	Resets DMA channel 0 interface adapter between FPGA Fabric and HPS DMA Controller	RW	0x1
22	emacptp	Resets EMAC PTP	RW	0x1
21	dmaocp	Resets DMA Controller ECC OCP DIagnostics modules.	RW	0x1
20	spis1	Resets SPIS1 controller	RW	0x1
19	spis0	Resets SPIS0 controller	RW	0x1
18	spim1	Resets SPIM1 controller	RW	0x1
17	spim0	Resets SPIM0 controller	RW	0x1
16	dma	Resets DMA controller	RW	0x1
15	sdmmcocp	Resets SDMMC ECC OCP DIagnostics modules.	RW	0x1
14	qspiocp	Resets QSPI ECC OCP DIagnostics modules.	RW	0x1
13	nandocp	Resets NAND ECC OCP DIagnostics modules.	RW	0x1
12	usb1ocp	Resets USB1 ECC OCP DIagnostics modules.	RW	0x1
11	usb0ocp	Resets USB0 ECC OCP DIagnostics modules.	RW	0x1
10	emac2ocp	Resets EMAC0 ECC OCP DIagnostics modules.	RW	0x1

Bit	Name	Description	Access	Reset
9	emac1ocp	Resets EMAC1 ECC OCP DIagnostics modules.	RW	0x1
8	emac0ocp	Resets EMAC0 ECC OCP DIagnostics modules.	RW	0x1
7	sdmmc	Resets SD/MMC controller	RW	0x1
6	qspi	Resets QSPI flash controller	RW	0x1
5	nand	Resets NAND flash controller	RW	0x1
4	usb1	Resets USB1	RW	0x1
3	usb0	Resets USB0	RW	0x1
2	emac2	Resets EMAC2	RW	0x1
1	emac1	Resets EMAC1	RW	0x1
0	emac0	Resets EMAC0	RW	0x1

per1modrst

The PER1MODRST register is used by software to trigger module resets for Slow Peripheral Group. Software explicitly asserts and de-asserts module reset signals by writing bits in the appropriate *MODRST register. It is up to software to ensure module reset signals are asserted for the appropriate length of time and are de-asserted in the correct order. It is also up to software to not assert a module reset signal that would prevent software from de-asserting the module reset signal. For example, software should not assert the module reset to the CPU executing the software.

Software writes a bit to 1 to assert the module reset signal and to 0 to de-assert the module reset signal.

All fields are reset by a cold reset.All fields are also reset by a warm reset if not masked by the corresponding PERWARMMASK field. The reset value of all fields is 1. This holds the corresponding module in reset until software is ready to release the module from reset by writing 0 to its field.

Module Instance	Base Address	Register Address
i_rst_mgr_rstmgr	0xFFD05000	0xFFD05028

Offset: 0x28

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved					gpio2 RW 0x1	gpio1 RW 0x1	gpio0 RW 0x1	Reserved						uart1 RW 0x1	uart0 RW 0x1
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			i2c4 RW 0x1	i2c3 RW 0x1	i2c2 RW 0x1	i2c1 RW 0x1	i2c0 RW 0x1	Reserved		sptimer1 RW 0x1	sptimer0 RW 0x1	l4systimer1 RW 0x1	l4systimer0 RW 0x1	watchdog1 RW 0x1	watchdog0 RW 0x1

per1modrst Fields

Bit	Name	Description	Access	Reset
26	gpio2	Resets GPIO2	RW	0x1
25	gpio1	Resets GPIO1	RW	0x1
24	gpio0	Resets GPIO0	RW	0x1
17	uart1	Resets UART1	RW	0x1
16	uart0	Resets UART0	RW	0x1
12	i2c4	Resets I2C4 controller	RW	0x1
11	i2c3	Resets I2C3 controller	RW	0x1
10	i2c2	Resets I2C2 controller	RW	0x1
9	i2c1	Resets I2C1 controller	RW	0x1
8	i2c0	Resets I2C0 controller	RW	0x1
5	sptimer1	Resets SP timer 1 connected to L4	RW	0x1
4	sptimer0	Resets SP timer 0 connected to L4	RW	0x1
3	l4systimer1	Resets l4sys_timer1	RW	0x1

Bit	Name	Description	Access	Reset
2	l4systimer0	Resets l4sys_timer0	RW	0x1
1	watchdog1	Resets Watchdog 1	RW	0x1
0	watchdog0	Resets Watchdog 0	RW	0x1

brgmodrst

The BRGMODRST register is used by software to trigger module resets (individual module reset signals). Software explicitly asserts and de-asserts module reset signals by writing bits in the appropriate *MODRST register. It is up to software to ensure module reset signals are asserted for the appropriate length of time and are de-asserted in the correct order. It is also up to software to not assert a module reset signal that would prevent software from de-asserting the module reset signal. For example, software should not assert the module reset to the CPU executing the software.

Software writes a bit to 1 to assert the module reset signal and to 0 to de-assert the module reset signal.

All fields are reset by a cold reset.All fields are also reset by a warm reset if not masked by the corresponding BRGWARMMASK field. The reset value of all fields is 1. This holds the corresponding module in reset until software is ready to release the module from reset by writing 0 to its field.

Module Instance	Base Address	Register Address
i_rst_mgr_rstmgr	0xFFD05000	0xFFD0502C

Offset: 0x2C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									ddrsch RW 0x1	f2ssdram2 RW 0x1	f2ssdram1 RW 0x1	f2ssdram0 RW 0x1	fpga2hps RW 0x1	lwhps2fpga RW 0x1	hps2fpga RW 0x1

brgmodrst Fields

Bit	Name	Description	Access	Reset
6	ddrsch	Resets logic in the DDR Scheduler in the NOC.	RW	0x1
5	f2ssdram2	Resets F2S_SDRAM2 Bridge	RW	0x1
4	f2ssdram1	Resets F2S_SDRAM1 Bridge	RW	0x1
3	f2ssdram0	Resets F2S_SDRAM0 Bridge	RW	0x1
2	fpga2hps	Resets FPGA2HPS Bridge	RW	0x1
1	lwhps2fpga	Resets LWHPS2FPGA Bridge	RW	0x1
0	hps2fpga	Resets HPS2FPGA Bridge	RW	0x1

sysmodrst

The SYSMODRST register is used by software to trigger module resets (individual module reset signals). Software explicitly asserts and de-asserts module reset signals by writing bits in the appropriate *MODRST register. It is up to software to ensure module reset signals are asserted for the appropriate length of time and are de-asserted in the correct order. It is also up to software to not assert a module reset signal that would prevent software from de-asserting the module reset signal. For example, software should not assert the module reset to the CPU executing the software.

Software writes a bit to 1 to assert the module reset signal and to 0 to de-assert the module reset signal.

All fields are only reset by a cold reset

Module Instance	Base Address	Register Address
i_rst_mgr_rstmgr	0xFFD05000	0xFFD05030

Offset: 0x30

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									ocramocp RW 0x0	sysdbg RW 0x0	s2f RW 0x0	fpgamgr RW 0x0	Reserved	ocram RW 0x0	rom RW 0x0

sysmodrst Fields

Bit	Name	Description	Access	Reset
6	ocramocp	Resets On-chip RAM ECC OCP Diagnostic module	RW	0x0
5	sysdbg	Resets logic that spans the system and debug domains.	RW	0x0
4	s2f	Resets logic in FPGA core that doesn't differentiate between HPS cold and warm resets	RW	0x0
3	fpgamgr	Resets FPGA Manager	RW	0x0
1	ocram	Resets On-chip RAM	RW	0x0
0	rom	Resets Boot ROM	RW	0x0

coldmodrst

The COLDMODRST register is used by software to trigger module resets (individual module reset signals). Software explicitly asserts and de-asserts module reset signals by writing bits in the appropriate *MODRST register. It is up to software to ensure module reset signals are asserted for the appropriate length of time and are

de-asserted in the correct order. It is also up to software to not assert a module reset signal that would prevent software from de-asserting the module reset signal. For example, software should not assert the module reset to the CPU executing the software.

Software writes a bit to 1 to assert the module reset signal and to 0 to de-assert the module reset signal.

All fields are only reset by a cold reset

Module Instance	Base Address	Register Address
i_rst_mgr_rstmgr	0xFFD05000	0xFFD05034

Offset: 0x34

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								iomgrcold RW 0x0	hmccold RW 0x0	tapcold RW 0x0	timestampcold RW 0x0	s2fcold RW 0x0	Reserved		clkmgrcold RW 0x0

coldmodrst Fields

Bit	Name	Description	Access	Reset
7	iomgrcold	Resets logic in IO Manager that is only reset by a cold reset (ignores warm reset)	RW	0x0
6	hmccold	Resets logic in HMC affected only by a cold reset.	RW	0x0
5	tapcold	Resets portion of DAP JTAG TAP controller no reset by a debug probe reset (i.e. nTRST pin). Cold reset only.	RW	0x0

Bit	Name	Description	Access	Reset
4	timestampcold	Resets debug timestamp to 0 (cold reset only)	RW	0x0
3	s2fcold	Resets logic in FPGA core that is only reset by a cold reset (ignores warm reset)	RW	0x0
0	clkmgrcold	Resets Clock Manager (cold reset only)	RW	0x0

nrstmodrst

The NRSTMODRST register is used by software to trigger module resets (individual module reset signals). Software explicitly asserts and de-asserts module reset signals by writing bits in the appropriate *MODRST register. It is up to software to ensure module reset signals are asserted for the appropriate length of time and are de-asserted in the correct order. It is also up to software to not assert a module reset signal that would prevent software from de-asserting the module reset signal. For example, software should not assert the module reset to the CPU executing the software.

Software writes a bit to 1 to assert the module reset signal and to 0 to de-assert the module reset signal.

All fields are only reset by a cold reset

Module Instance	Base Address	Register Address
i_rst_mgr_rstmgr	0xFFD05000	0xFFD05038

Offset: 0x38

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															nrstpinoe RW 0x0

nrstmodrst Fields

Bit	Name	Description	Access	Reset
0	nrstpinoe	Drives NRST output enable with data tied low to extend cold and warm reset	RW	0x0

dbgmodrst

The DBGMODRST register is used by software to trigger module resets (individual module reset signals). Software explicitly asserts and de-asserts module reset signals by writing bits in the appropriate *MODRST register. It is up to software to ensure module reset signals are asserted for the appropriate length of time and are de-asserted in the correct order. It is also up to software to not assert a module reset signal that would prevent software from de-asserting the module reset signal. For example, software should not assert the module reset to the CPU executing the software.

Software writes a bit to 1 to assert the module reset signal and to 0 to de-assert the module reset signal.

All fields are only reset by a cold reset

Module Instance	Base Address	Register Address
i_rst_mgr_rstmgr	0xFFD05000	0xFFD0503C

Offset: 0x3C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															dbg RW 0x0

dbgmodrst Fields

Bit	Name	Description	Access	Reset
0	dbg	Resets logic located only in the debug domain.	RW	0x0

mpuwarmmask

The MPUWARMMASK register is used by software to mask the assertion of module reset signals for hardware sequenced warm resets. There is a writeable bit for each module reset signal that is asserted by default on a hardware sequenced warm reset. If the bit is 1, the module reset signal is asserted by a hardware sequenced warm reset. If the bit is 0, the module reset signal is not changed by a hardware sequenced warm reset. The bit assignments of the *WARMMASK registers match the corresponding *MODRST registers. Any module reset signals that are never asserted by a warm reset have reserved bit offsets and are tied to 0 (read as 0, writes are ignored).

All fields are only reset by a cold reset.

Module Instance	Base Address	Register Address
i_rst_mgr_rstmgr	0xFFD05000	0xFFD05040

Offset: 0x40

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															wds RW 0x1

mpuwarmmask Fields

Bit	Name	Description	Access	Reset
0	wds	Masks hardware sequenced warm reset for both per-CPU Watchdog Reset Status registers in MPU	RW	0x1

per0warmmask

The PER0WARMMASK register is used by software to mask the assertion of Peripheral and Fast Peripheral reset signals for hardware sequenced warm resets. There is a writeable bit for each module reset signal that is asserted by default on a hardware sequenced warm reset. If the bit is 1, the module reset signal is asserted by a hardware sequenced warm reset. If the bit is 0, the module reset signal is not changed by a hardware sequenced warm reset. The bit assignments of the *WARMMASK registers match the corresponding *MODRST registers. Any module reset signals that are never asserted by a warm reset have reserved bit offsets and are tied to 0 (read as 0, writes are ignored).

All fields are only reset by a cold reset.

Module Instance	Base Address	Register Address
i_rst_mgr_rstmgr	0xFFD05000	0xFFD05044

Offset: 0x44

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dmaif7 RW 0x1	dmaif6 RW 0x1	dmaif5 RW 0x1	dmaif4 RW 0x1	dmaif3 RW 0x1	dmaif2 RW 0x1	dmaif1 RW 0x1	dmaif0 RW 0x1	Reserved	emacptp RW 0x1	dmaocp RW 0x1	spis1 RW 0x1	spis0 RW 0x1	spim1 RW 0x1	spim0 RW 0x1	dma RW 0x1
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
sdmmcocp RW 0x1	qspiocp RW 0x1	nandocp RW 0x1	usb1ocp RW 0x1	usb0ocp RW 0x1	emac2ocp RW 0x1	emac1ocp RW 0x1	emac0ocp RW 0x1	sdmmc RW 0x1	qspi RW 0x1	nand RW 0x1	usb1 RW 0x1	usb0 RW 0x1	emac2 RW 0x1	emac1 RW 0x1	emac0 RW 0x1

per0warmmask Fields

Bit	Name	Description	Access	Reset
31	dmaif7	Masks hardware sequenced warm reset for DMA channel 7 interface adapter between FPGA Fabric and HPS DMA Controller	RW	0x1
30	dmaif6	Masks hardware sequenced warm reset for DMA channel 6 interface adapter between FPGA Fabric and HPS DMA Controller	RW	0x1
29	dmaif5	Masks hardware sequenced warm reset for DMA channel 5 interface adapter between FPGA Fabric and HPS DMA Controller	RW	0x1
28	dmaif4	Masks hardware sequenced warm reset for DMA channel 4 interface adapter between FPGA Fabric and HPS DMA Controller	RW	0x1
27	dmaif3	Masks hardware sequenced warm reset for DMA channel 3 interface adapter between FPGA Fabric and HPS DMA Controller	RW	0x1
26	dmaif2	Masks hardware sequenced warm reset for DMA channel 2 interface adapter between FPGA Fabric and HPS DMA Controller	RW	0x1
25	dmaif1	Masks hardware sequenced warm reset for DMA channel 1 interface adapter between FPGA Fabric and HPS DMA Controller	RW	0x1
24	dmaif0	Masks hardware sequenced warm reset for DMA channel 0 interface adapter between FPGA Fabric and HPS DMA Controller	RW	0x1
22	emacptp	Masks hardware sequenced warm reset for EMAC PTP	RW	0x1
21	dmaocp	Masks hardware sequenced warm reset for DMA Controller ECC OCP DIagnos-tics module.	RW	0x1

Bit	Name	Description	Access	Reset
20	spis1	Masks hardware sequenced warm reset for SPIS1 controller	RW	0x1
19	spis0	Masks hardware sequenced warm reset for SPIS0 controller	RW	0x1
18	spim1	Masks hardware sequenced warm reset for SPIM1 controller	RW	0x1
17	spim0	Masks hardware sequenced warm reset for SPIM0 controller	RW	0x1
16	dma	Masks hardware sequenced warm reset for DMA controller	RW	0x1
15	sdmmcocp	Masks hardware sequenced warm reset for SDMMC ECC OCP DIagnostics module.	RW	0x1
14	qspiocp	Masks hardware sequenced warm reset for QSPI ECC OCP DIagnostics module.	RW	0x1
13	nandocp	Masks hardware sequenced warm reset for NAND ECC OCP DIagnostics modules.	RW	0x1
12	usb1ocp	Masks hardware sequenced warm reset for USB1 ECC OCP DIagnostics module.	RW	0x1
11	usb0ocp	Masks hardware sequenced warm reset for USB0 ECC OCP DIagnostics module.	RW	0x1
10	emac2ocp	Masks hardware sequenced warm reset for EMAC2 ECC OCP DIagnostics modules.	RW	0x1
9	emac1ocp	Masks hardware sequenced warm reset for EMAC1 ECC OCP DIagnostics modules.	RW	0x1
8	emac0ocp	Masks hardware sequenced warm reset for EMAC0 ECC OCP DIagnostics modules.	RW	0x1
7	sdmmc	Masks hardware sequenced warm reset for SD/MMC controller	RW	0x1
6	qspi	Masks hardware sequenced warm reset for QSPI flash controller	RW	0x1

Bit	Name	Description	Access	Reset
5	nand	Masks hardware sequenced warm reset for NAND flash controller	RW	0x1
4	usb1	Masks hardware sequenced warm reset for USB1USB1	RW	0x1
3	usb0	Masks hardware sequenced warm reset for USB0	RW	0x1
2	emac2	Masks hardware sequenced warm reset for EMAC2	RW	0x1
1	emac1	Masks hardware sequenced warm reset for EMAC1	RW	0x1
0	emac0	Masks hardware sequenced warm reset for EMAC0	RW	0x1

per1warmmask

The PER1WARMMASK register is used by software to mask the assertion of module reset signals for hardware sequenced warm resets. There is a writeable bit for each module reset signal that is asserted by default on a hardware sequenced warm reset. If the bit is 1, the module reset signal is asserted by a hardware sequenced warm reset. If the bit is 0, the module reset signal is not changed by a hardware sequenced warm reset. The bit assignments of the *WARMMASK registers match the corresponding *MODRST registers. Any module reset signals that are never asserted by a warm reset have reserved bit offsets and are tied to 0 (read as 0, writes are ignored).

All fields are only reset by a cold reset.

Module Instance	Base Address	Register Address
i_rst_mgr_rstmgr	0xFFD05000	0xFFD05048

Offset: 0x48

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved					gpio2 RW 0x1	gpio1 RW 0x1	gpio0 RW 0x1	Reserved						uart1 RW 0x1	uart0 RW 0x1
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			i2c4 RW 0x1	i2c3 RW 0x1	i2c2 RW 0x1	i2c1 RW 0x1	i2c0 RW 0x1	Reserved		sptimer1 RW 0x1	sptimer0 RW 0x1	l4systimer1 RW 0x1	l4systimer0 RW 0x1	watchdog1 RW 0x1	watchdog0 RW 0x1

per1warmmask Fields

Bit	Name	Description	Access	Reset
26	gpio2	Masks hardware sequenced warm reset for GPIO2	RW	0x1
25	gpio1	Masks hardware sequenced warm reset for GPIO1	RW	0x1
24	gpio0	Masks hardware sequenced warm reset for GPIO0	RW	0x1
17	uart1	Masks hardware sequenced warm reset for UART1	RW	0x1
16	uart0	Masks hardware sequenced warm reset forUART0	RW	0x1
12	i2c4	Masks hardware sequenced warm reset for I2C4 controller	RW	0x1
11	i2c3	Masks hardware sequenced warm reset for I2C3 controller	RW	0x1
10	i2c2	Masks hardware sequenced warm reset for I2C2 controller	RW	0x1
9	i2c1	Masks hardware sequenced warm reset for 2C1 controller	RW	0x1
8	i2c0	Masks hardware sequenced warm reset for 2C0 controller	RW	0x1
5	sptimer1	Masks hardware sequenced warm reset for SP timer 1 connected to L4	RW	0x1

Bit	Name	Description	Access	Reset
4	sptimer0	Masks hardware sequenced warm reset for SP timer 0 connected to L4	RW	0x1
3	l4systimer1	Masks hardware sequenced warm reset for l4sys_timer1	RW	0x1
2	l4systimer0	Masks hardware sequenced warm reset for l4sys_timer0	RW	0x1
1	watchdog1	Masks hardware sequenced warm reset for Watchdog 1	RW	0x1
0	watchdog0	Masks hardware sequenced warm reset for Watchdog 0	RW	0x1

brgwarmmask

The Bridge_WARM_MASK register is used by software to mask the assertion of module reset signals for hardware sequenced warm resets. There is a writeable bit for each module reset signal that is asserted by default on a hardware sequenced warm reset. If the bit is 1, the module reset signal is asserted by a hardware sequenced warm reset. If the bit is 0, the module reset signal is not changed by a hardware sequenced warm reset. The bit assignments of the *WARMMASK registers match the corresponding *MODRST registers. Any module reset signals that are never asserted by a warm reset have reserved bit offsets and are tied to 0 (read as 0, writes are ignored).

All fields are only reset by a cold reset.

Module Instance	Base Address	Register Address
i_rst_mgr_rstmgr	0xFFD05000	0xFFD0504C

Offset: 0x4C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									ddrsch RW 0x1	f2ssdram2 RW 0x1	f2ssdram1 RW 0x1	f2ssdram0 RW 0x1	fpga2hps RW 0x1	lwhps2fpga RW 0x1	hps2fpga RW 0x1

brgwarmmask Fields

Bit	Name	Description	Access	Reset
6	ddrsch	Masks hardware sequenced warm reset for the DDR Scheduler in the NOC.	RW	0x1
5	f2ssdram2	Masks hardware sequenced warm reset for F2S_SDRAM2 Bridge	RW	0x1
4	f2ssdram1	Masks hardware sequenced warm reset for F2S_SDRAM1 Bridge	RW	0x1
3	f2ssdram0	Masks hardware sequenced warm reset for F2S_SDRAM0 Bridge	RW	0x1
2	fpga2hps	Masks hardware sequenced warm reset for FPGA2HPS Bridge	RW	0x1
1	lwhps2fpga	Masks hardware sequenced warm reset for LWHPS2FPGA Bridge	RW	0x1
0	hps2fpga	Masks hardware sequenced warm reset for HPS2FPGA Bridge	RW	0x1

syswarmmask

The SYSWARMMASK register is used by software to mask the assertion of module reset signals for hardware sequenced warm resets. There is a writeable bit for each module reset signal that is asserted by default on a hardware sequenced warm reset. If the bit is 1, the module reset signal is asserted by a hardware sequenced warm reset. If the bit is 0, the module reset signal is not changed by a hardware sequenced warm reset. The bit assignments of the *WARMMASK registers match the corresponding *MODRST registers. Any module reset signals that are never asserted by a warm reset have reserved bit offsets and are tied to 0 (read as 0, writes are ignored).

All fields are only reset by a cold reset.
Fields in the SYSMODRST register associated with cold reset or debug

domain reset aren't present in the MISCWARMMASK register and are reserved.

Module Instance	Base Address	Register Address
i_rst_mgr_rstmgr	0xFFD05000	0xFFD05050

Offset: 0x50

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									ocramocp RW 0x1	sysdbg RW 0x1	s2f RW 0x1	fpgamgr RW 0x1	Reserved	ocram RW 0x1	rom RW 0x1

syswarmmask Fields

Bit	Name	Description	Access	Reset
6	ocramocp	Masks hardware sequenced warm reset for On-chip RAM ECC OCP Diagnostic module	RW	0x1
5	sysdbg	Masks hardware sequenced warm reset for logic that spans the system and debug domains.	RW	0x1
4	s2f	Masks hardware sequenced warm reset for logic in FPGA core that doesn't differentiate between HPS cold and warm resets	RW	0x1
3	fpgamgr	Masks hardware sequenced warm reset for FPGA Manager	RW	0x1
1	ocram	Masks hardware sequenced warm reset for On-chip RAM	RW	0x1

Bit	Name	Description	Access	Reset
0	rom	Masks hardware sequenced warm reset for Boot ROM	RW	0x1

nrstwarmmask

The NRSTWARMMASK register is used by software to mask the assertion of nRST for hardware sequenced warm resets. There is a writeable bit that is asserted by default on a hardware sequenced warm reset. If the bit is 1, the module reset signal is asserted by a hardware sequenced warm reset. If the bit is 0, the module reset signal is not changed by a hardware sequenced warm reset. The bit assignments of the *WARMMASK registers match the corresponding *MODRST registers. Any module reset signals that are never asserted by a warm reset have reserved bit offsets and are tied to 0 (read as 0, writes are ignored).
Only reset by a cold reset.

Module Instance	Base Address	Register Address
i_rst_mgr_rstmgr	0xFFD05000	0xFFD05054

Offset: 0x54

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															nrstpinoe RW 0x1

nrstwarmmask Fields

Bit	Name	Description	Access	Reset
0	nrstpinoe	Masks hardware sequenced warm reset for nrst_pin_oe	RW	0x1

l3warmmask

The L3WARMMASK register is used by software to mask the assertion of module reset signals for hardware sequenced warm resets. There is a writeable bit for each module reset signal that is asserted by default on a hardware sequenced warm reset. If the bit is 1, the module reset signal is asserted by a hardware sequenced warm reset. If the bit is 0, the module reset signal is not changed by a hardware sequenced warm reset. The bit assignments of the *WARMMASK registers match the corresponding *MODRST registers. Any module reset signals that are never asserted by a warm reset have reserved bit offsets and are tied to 0 (read as 0, writes are ignored).

All fields are only reset by a cold reset.

Note that there is no module reset for the L3 interconnect in any *MODRST register because there would be no way for a master to remove the module reset to the L3 once asserted.

Module Instance	Base Address	Register Address
i_rst_mgr_rstmgr	0xFFD05000	0xFFD05058

Offset: 0x58

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															l3 RW 0x1

l3warmmask Fields

Bit	Name	Description	Access	Reset
0	l3	Masks hardware sequenced warm reset to L3 Interconnect	RW	0x1

tststa

status fields used for testing the Reset Manager.

Module Instance	Base Address	Register Address
i_rst_mgr_rstmgr	0xFFD05000	0xFFD0505C

Offset: 0x5C

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									dbgrstst RO 0x0			warmrstst RO 0x0			

tststa Fields

Bit	Name	Description	Access	Reset
6:4	dbgrstst	debug reset control FSM state	RO	0x0
3:0	warmrstst	warm reset control FSM state	RO	0x0

tstscratch

SW can write/read this register without side effect to reset manager function.

Module Instance	Base Address	Register Address
i_rst_mgr_rstmgr	0xFFD05000	0xFFD05060

Offset: 0x60

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
fld0 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
fld0 RW 0x0															

tstscratch Fields

Bit	Name	Description	Access	Reset
31:0	fld0	field for SW write/read	RW	0x0

hdsktimeout

The Warm Reset handshake timeout will default to 10,240 which at 100 MHz for l4_sys_free_clk will 102.4 micro-seconds. This value will be a 25 bit programmable value in SW. The reason for this is the HMC adaptor may need a longer time to clear all outstanding SDRAM transactions. The maximum programmable value would be 20.97 msec

Module Instance	Base Address	Register Address
i_rst_mgr_rstmgr	0xFFD05000	0xFFD05064

Offset: 0x64

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							val RW 0x2800								
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
val RW 0x2800															

hdsktimeout Fields

Bit	Name	Description	Access	Reset
24:0	val	hand shake timeout	RW	0x2800

hmcintr

HMC GPIO Interrupt

Module Instance	Base Address	Register Address
i_rst_mgr_rstmgr	0xFFD05000	0xFFD05068

Offset: 0x68

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															intr RW 0x0

hmcintr Fields

Bit	Name	Description	Access	Reset
0	intr	Interrupt	RW	0x0

hmcintren

HMC Interrupt Enable

Module Instance	Base Address	Register Address
i_rst_mgr_rstmgr	0xFFD05000	0xFFD0506C

Offset: 0x6C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															en RW 0x0

hmcintren Fields

Bit	Name	Description	Access	Reset
0	en	HMC Interrupt Enable Signal for hmcintr	RW	0x0

hmcintrens

HMC Interrupt Enable Set

Module Instance	Base Address	Register Address
i_rst_mgr_rstmgr	0xFFD05000	0xFFD05070

Offset: 0x70

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															en RW 0x0

hmcintrens Fields

Bit	Name	Description	Access	Reset
0	en	Interrupt Enable Set Signal	RW	0x0

hmcintrenr

HMC Interrupt Enable Clear

Module Instance	Base Address	Register Address
i_rst_mgr_rstmgr	0xFFD05000	0xFFD05074

Offset: 0x74

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															en RW 0x0

hmcintrenr Fields

Bit	Name	Description	Access	Reset
0	en	HMC Interrupt Enable Clear Signal	RW	0x0

hmcgpout

GPIO output for HMC and corresponding interrupt

Module Instance	Base Address	Register Address
i_rst_mgr_rstmgr	0xFFD05000	0xFFD05078

Offset: 0x78

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								out RW 0x0							

hmcgpout Fields

Bit	Name	Description	Access	Reset
7:0	out	hmc gpio out	RW	0x0

hmcgpin

GPIO input for HMC and corresponding interrupt

Module Instance	Base Address	Register Address
i_rst_mgr_rstmgr	0xFFD05000	0xFFD0507C

Offset: 0x7C

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								in RO 0x0							

hmcgpin Fields

Bit	Name	Description	Access	Reset
7:0	in	hmc gpio in	RO	0x0

Document Revision History

Table 3-13: Document Revision History

Date	Version	Changes
October 2016	2016.10.28	Maintenance release.
May 2016	2016.05.03	Maintenance release.
November 2015	2015.11.02	Updated "Reset Pins" section.
May 2015	2015.05.04	"Slave Interface" and "Status Register" sections updated.
December 2014	2014.12.15	Maintenance release.
August 2014	2014.08.18	Initial release.

FPGA Manager 4

2016.10.28

a10_5v4 Subscribe Send Feedback

The FPGA manager in the hard processor system (HPS) manages and monitors the FPGA portion of the system on a chip (SoC) device. The FPGA manager can configure the FPGA fabric from the HPS, monitor the state of the FPGA, and drive or sample signals to or from the FPGA fabric.

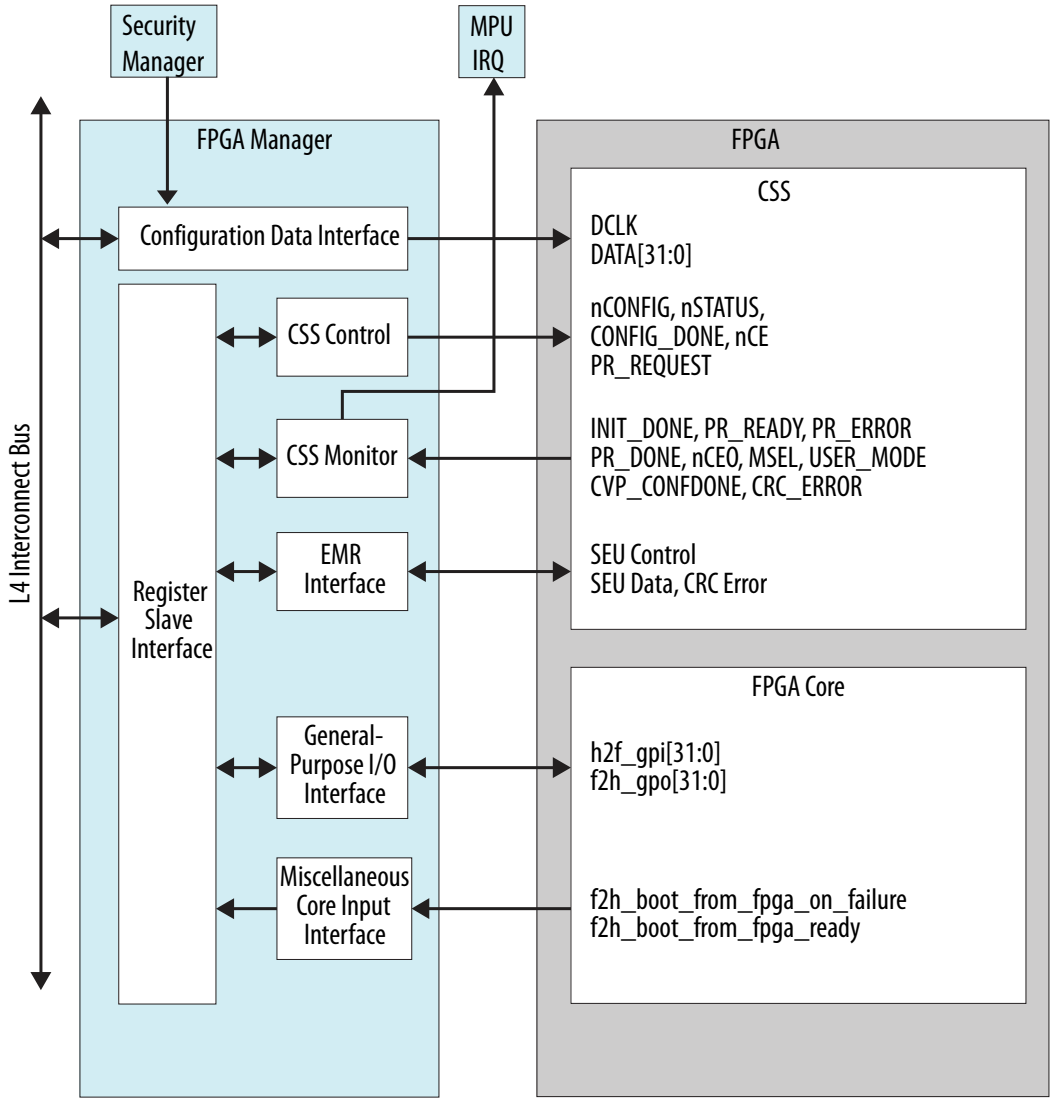
Features of the FPGA Manager

The FPGA manager provides the following functionality and features:

- Full configuration and partial reconfiguration
- CRC error message extraction
- General purpose I/O signals to the FPGA
- Boot from FPGA control
- Generation of interrupts based on FPGA status changes

FPGA Manager Block Diagram and System Integration

Figure 4-1: FPGA Manager Block Diagram



The FPGA Manager consists of the following blocks:

- Configuration Data Interface - Accepts and transfers the configuration and decryption data to the FPGA configuration sub system (CSS)
- Register Slave Interface - Accesses the control and status registers in the FPGA Manager
- CSS Control - Controls full and partial configuration of the FPGA
- CSS Monitor - Monitors the status of the FPGA during a full or partial configuration of the device

- Error Message Register (EMR) Interface - Error message extraction, in case of CRC errors in the FPGA
- General Purpose I/O Interface - Receives and drives 32 bits of general purpose I/O to and from the FPGA
- Miscellaneous Core Input Interface - Receives control input signals from the FPGA to support booting the HPS from the FPGA

Functional Description of the FPGA Manager

FPGA Configuration

You can configure the FPGA using an external device or through the HPS. This section highlights configuring the FPGA through the HPS.

The FPGA manager connects to the configuration logic in the FPGA portion of the device using a mode similar to how external logic (for example, MAX II or an intelligent host) configures the FPGA in fast passive parallel (FPP) mode. FPGA configuration through the HPS supports all the capabilities of FPP mode, including the following items:

- FPGA configuration
- Partial FPGA reconfiguration
- Decompression
- Advanced Encryption Standard (AES) encryption

Note: The FPGA manager supports a data width of 16 or 32 bits. When configuring the FPGA fabric from the HPS, Altera recommends that you always set the data width to 32 bits. For partial reconfiguration, the 16-bit and 32-bit data widths are options.

Table 4-1: MSEL Pin Settings for Each Configuration Scheme of Arria 10 Devices

Configuration Scheme	V_{CCPGM}(V)	Power-On Reset (POR) Delay	Valid MSEL[2:0]
FPP (x16 and x32)	1.8	Fast	000
		Standard	001

The FPGA Manager can be configured to accept configuration data directly from the MPU or the DMA engine. Either the processor or the DMA engine moves data from memory to the FPGA Manager data image register space `img_data_w`.

Configuration data is buffered in a 64 deep x 32 bits wide FIFO in the FPGA Manager. Status information such as FIFO empty/full and remaining depth can be accessed through a status register `imgcfg_stat`. FIFO empty/full conditions can also be enabled to generate an interrupt.

If using the DMA engine to move FPGA configuration data to the FPGA Manager, a FIFO threshold value can be set in the `dma_config` register. This value is used to control the assertion of a DMA transfer request from the FPGA Manager to the DMA engine.

Before sending FPGA configuration data to the FPGA Manager HPS, software sets the clock-to-data ratio field (`CDRATIO`) and configuration data width bit (`CFGWIDTH`) in the image control 2 register (`imgcfg_ctrl_02`).

Related Information

- **Configuration, Design Security, and Remote System Upgrades in Arria 10 Devices**
  For more information about configuring the FPGA through the HPS, refer to the "Configuration, Design Security, and Remote System Upgrade in Arria 10 Devices" appendix in the *Arria 10 Core Fabric and General Purpose I/Os Handbook*.
- **Booting and Configuration** on page 30-1

FPGA Status

Configuration signals from the FPGA CSS such as `INIT_DONE`, `CRC_ERROR` and `PR_DONE` are monitored by the FPGA Manager. Software configures the monitor block through the register slave interface, and can either poll FPGA signals or be interrupted. Monitored signals can be read through the `imgcfg_stat` register as well as the `intr_masked_status` register. Each of the monitored signals can generate an interrupt to the MPU global interrupt controller. Each interrupt source can be enabled and the polarity of the signals generating the interrupt can also be selected through the `intr_mask` and `intr_polarity` registers in the FPGA Manager.

Error Message Extraction

Cyclic redundancy check (CRC) errors from the FPGA fabric are monitored by the FPGA Manager. Upon assertion of a CRC error signal from the FPGA, the FPGA Manager extracts information about the error including:

- Error syndrome
- Error location
- Error type

A CRC error interrupt from the FPGA manager can be enabled through software. Software can then extract the CRC error information from the error message register (EMR) data interface. The number of valid error information bits in the EMR data registers depends on the specific FPGA device.

Data AES Decryption

The configuration data interface can also be used to transmit data from the HPS to the FPGA CSS Advanced Encryption Standard (AES) decryption engine. Software should sample the status registers in FPGA manager before switching modes between FPGA configuration and AES decryption. The FPGA manager does not do anything specific to handle a situation where software starts an AES decryption while in the middle of an FPGA configuration initiated by HPS, and the end result in this case is undefined.

Boot Handshake

There are two input signals from the FPGA to control HPS boot from the FPGA. Both are synchronized within the FPGA Manager. Boot software reads these signals before accessing a boot image in the FPGA. The following table describes the functionality of these signals.

Signal	Description
f2h_boot_from_fpga_on_failure	Indicates whether a fallback second-stage boot loader image is available in the FPGA on-chip RAM at memory location 0x0. The fallback second-stage boot loader image is used only if the HPS boot ROM does not find a valid second-stage boot loader image in the selected flash memory device.
f2h_boot_from_fpga_ready	The f2h_boot_from_fpga_ready signal is used by the Boot ROM when accessing the public key stored in the FPGA. The Boot ROM will only use the signal if asserted to boot with the public key. Indicates a second-stage boot loader image is available in an FPGA on-chip RAM at memory location 0x0 and it is ready to be accessed.

General Purpose I/O

Thirty-two general purpose inputs and thirty-two general purpose outputs are provided to the FPGA and are controlled through registers in the FPGA Manager.

No interrupts are generated through the input pins. All inputs are synchronized within the FPGA Manager. Output signals should be synchronized in the FPGA.

Clock

The FPGA manager has two clock input signals which are asynchronous to each other. The clock manager generates these two clocks:

- cfg_clk—the configuration slave interface clock input and also the DCLK output reference for FPGA configuration. Enable this clock in the clock manager only when configuration is active or when the configuration slave interface needs to respond to master requests.
- l4_mp_clk—the register slave interface clock.

Related Information

Clock Manager on page 2-1

Reset

The FPGA manager has the fpga_manager_rst_n reset signal. The reset manager drives this signal to the FPGA manager on a cold or warm reset. All distributed reset signals in the FPGA manager are asserted asynchronously at the same time and deasserted synchronously to their associated clocks.

Related Information

Reset Manager on page 3-1

FPGA Manager Address Map and Register Definitions

For complete HPS address map and register definitions, refer to the **Arria 10 HPS Address Map and Register Definitions**.

fpga_mgr_fpgamgrdata Address Map

Module Instance	Base Address	End Address
i_fpga_mgr_fpgamgrdata	0xFFCFE400	0xFFCFFFFF

Register	Offset	Width	Access	Reset Value	Description
img_data_w on page 4-6	0x	32	WO	0x0	Write Data Register

fpga_mgr_fpgamgrdata Summary

Base Address: 0xFFCFE400

Register Address Offset	Bit Fields															
i_fpga_mgr_fpgamgrdata																
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
img_data_w 0x0	data RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	data RW 0x0															

img_data_w

Used to send configuration image to FPGA.

Module Instance	Base Address	Register Address
i_fpga_mgr_fpgamgrdata	0xFFCFE400	0xFFCFE400

Offset: 0x0

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
data RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
data RW 0x0															

img_data_w Fields

Bit	Name	Description	Access	Reset
31:0	data	Accepts configuration image to be sent to the CSS when the HPS configures the FPGA. Software normally writes the register.	RW	0x0

fpga_mgr_fpgamgrregs Address Map

Module Instance	Base Address	End Address
i_fpga_mgr_fpgamgrregs	0xFFD03000	0xFFD03FFF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
dclkcnt on page 4-16	0x8	32	RW	0x0	DCLK Count Register
dclkstat on page 4-17	0xC	32	RW	0x0	DCLK Status Register
gpo on page 4-18	0x10	32	RW	0x0	General-Purpose Output Register
gpi on page 4-19	0x14	32	RO	0x0	General-Purpose Input Register
misci on page 4-19	0x18	32	RO	0x0	Miscellaneous Input Register

Register	Offset	Width	Access	Reset Value	Description
emr_data0 on page 4-21	0x30	32	RO	0x0	Extracted EMR register content
emr_data1 on page 4-21	0x34	32	RO	0x0	Extracted EMR register content
emr_data2 on page 4-22	0x38	32	RO	0x	bits [95:64] of EMR register
emr_data3 on page 4-23	0x3C	32	RO	0x	bits [119:96] of EMR register
emr_data4 on page 4-23	0x40	32	RO	0x	bits [159:128] of EMR register
emr_data5 on page 4-24	0x44	32	RO	0x	bits [171:160] of EMR register
emr_valid on page 4-25	0x48	32	RW	0x0	
emr_en on page 4-25	0x4C	32	RW	0x77000000	
jtag_config on page 4-26	0x50	32	RW	0x1400	Scan-Chain Enable Register
jtag_status on page 4-29	0x54	32	RO	0x500	Control/Status Word Register
jtag_kick on page 4-31	0x58	32	WO	0x0	TCK Divide ratio
jtag_data_w on page 4-32	0x60	32	WO	0x0	TX FIFO Write
jtag_data_r on page 4-33	0x64	32	RO	0x0	RX FIFO Read
imgcfg_ctrl_00 on page 4-34	0x70	32	RW	0x107	
imgcfg_ctrl_01 on page 4-36	0x74	32	RW	0x1000001	
imgcfg_ctrl_02 on page 4-37	0x78	32	RW	0x200	Control Register

Register	Offset	Width	Access	Reset Value	Description
imgcfg_stat on page 4-39	0x80	32	RO	0x1000000	This is the unmasked status. Value of corresponding inputs from CSS or PINs, without considering the intr_mask or intr_polarity.
intr_masked_status on page 4-42	0x84	32	RW	0x0	When you read this register you read the active high pending interrupt status of corresponding bit. This value is after the masking specified by intr_mask and after the polarity conversion as specified in intr_polarity
intr_mask on page 4-44	0x88	32	RW	0x33073FFF	Mask for interrupts. A value of 1 in a particular bit will cause the specific interrupt to be masked.
intr_polarity on page 4-45	0x8C	32	RW	0x33073FFF	Active Level of the signal to generate interrupt. 0 :Active LOW. An interrupt will be generated when that particular bit/signal is LOW 1: Active HIGH. An interrupt will be generated when that particular bit/signal is HIGH
dma_config on page 4-47	0x90	32	RW	0x0	Control/Status Word Register
imgcfg_fifo_status on page 4-48	0x94	32	RO	0x200	Control/Status Word Register

fpga_mgr_fpgamgrregs Summary

Base Address: 0xFFD03000

Register Address Offset	Bit Fields															
i_fpga_mgr_fpgamgrregs																
dclkcnt 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RW 0x0															
dclkstat 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															dcntdone RW 0x0
gpo 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	value RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	value RW 0x0															
gpi 0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	value RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	value RO 0x0															
misci 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														bootFPGA rdy RO 0x0	bootFPGA fail RO 0x0

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
emr_data0 0x30	value RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	value RO 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
emr_data1 0x34	value RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	value RO 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
emr_data2 0x38	value RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	value RO 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
emr_data3 0x3C	value RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	value RO 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
emr_data4 0x40	value RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	value RO 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
emr_data5 0x44	value RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	value RO 0x0															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
emr_valid 0x48	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															vld 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
emr_en 0x4C	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															en RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
jtag_config 0x50	txSize RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	tckRatio RW 0x14								Reserved			trstEn RW 0x0	Reserved	loopBackEn RW 0x0	jtagPortEn RW 0x0	JtagHostEn RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
jtag_status 0x54	txDoneSize 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SessionStatus 0x0	Reserved			rxFifoFull 0x0	rxFifoEmpty 0x1	txFifoFull 0x0	txFifoEmpty 0x1	rxFifoLevel 0x0				txFifoLevel 0x0			
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
jtag_kick 0x58	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved												clearRxFifo 0x0	clearTxFifo 0x0	stopSession 0x0	startSession 0x0

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
jtag_data_w 0x60	tmsData 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	tdiData 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
jtag_data_r 0x64	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	tdiData 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
imgcfg_ctrl_00 0x70	Reserved							s2f_condone_oe RW 0x0	Reserved							s2f_nstatus_oe RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							s2f_nconfig RW 0x1	Reserved					s2f_nenable_condone RW 0x1	s2f_nenable_nstatus RW 0x1	s2f_nenable_nconfig RW 0x1
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
imgcfg_ctrl_01 0x74	Reserved							s2f_nce RW 0x1	Reserved							s2f_pr_request RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															s2f_nenable_config RW 0x1

Register Address Offset	Bit Fields															
imgcfg_ctrl_02 0x78	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved							cfgwidth RW 0x0	Reserved						cdratio RW 0x0	
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							en_cfg_data RW 0x0	Reserved							en_cfg_ctrl 0x0
imgcfg_stat 0x80	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved		emr RO 0x0	jtagm RO 0x0	Reserved		imgcfg_FifoFull RO 0x0	imgcfg_FifoEmpty RO 0x1	Reserved					f2s_msel2 RO 0x0	f2s_msel1 RO 0x0	f2s_msel0 RO 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		f2s_nceo_oe RO 0x0	f2s_nconfig_pin RO 0x0	f2s_pr_error RO 0x0	f2s_pr_done RO 0x0	f2s_pr_ready RO 0x0	f2s_cvp_conf_done RO 0x0	f2s_condone_oe RO 0x0	f2s_condone_pin RO 0x0	f2s_nstatus_oe RO 0x0	f2s_nstatus_pin RO 0x0	f2s_initdone_oe RO 0x0	f2s_usermode RO 0x0	f2s_early_usermode RO 0x0	f2s_crc_error RO 0x0
intr_masked_status 0x84	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved		emr RW 0x0	jtagm RW 0x0	Reserved		imgcfg_FifoFull RW 0x0	imgcfg_FifoEmpty RW 0x0	Reserved					f2s_msel2 RW 0x0	f2s_msel1 RW 0x0	f2s_msel0 RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		f2s_nceo_oe RW 0x0	f2s_nconfig_pin RW 0x0	f2s_pr_error RW 0x0	f2s_pr_done RW 0x0	f2s_pr_ready RW 0x0	f2s_cvp_conf_done RW 0x0	f2s_condone_oe RW 0x0	f2s_condone_pin RW 0x0	f2s_nstatus_oe RW 0x0	f2s_nstatus_pin RW 0x0	f2s_initdone_oe RW 0x0	f2s_usermode RW 0x0	f2s_early_usermode RW 0x0	f2s_crc_error RW 0x0

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
intr_mask 0x88	Reserved		emr RW 0x1	jtagm RW 0x1	Reserved		imgcfg_FifoFull RW 0x1	imgcfg_FifoEmpty RW 0x1	Reserved					f2s_msel2 RW 0x1	f2s_msel1 RW 0x1	f2s_msel0 RW 0x1
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		f2s_nceo_oe RW 0x1	f2s_nconfig_pin RW 0x1	f2s_pr_error RW 0x1	f2s_pr_done RW 0x1	f2s_pr_ready RW 0x1	f2s_cvp_conf_done RW 0x1	f2s_condone_oe RW 0x1	f2s_condone_pin RW 0x1	f2s_nstatus_oe RW 0x1	f2s_nstatus_pin RW 0x1	f2s_initdone_oe RW 0x1	f2s_usermode RW 0x1	f2s_early_usermode RW 0x1	f2s_crc_error RW 0x1
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
intr_polarity 0x8C	Reserved		emr RW 0x1	jtagm RW 0x1	Reserved		imgcfg_FifoFull RW 0x1	imgcfg_FifoEmpty RW 0x1	Reserved					f2s_msel2 RW 0x1	f2s_msel1 RW 0x1	f2s_msel0 RW 0x1
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		f2s_nceo_oe RW 0x1	f2s_nconfig_pin RW 0x1	f2s_pr_error RW 0x1	f2s_pr_done RW 0x1	f2s_pr_ready RW 0x1	f2s_cvp_conf_done RW 0x1	f2s_condone_oe RW 0x1	f2s_condone_pin RW 0x1	f2s_nstatus_oe RW 0x1	f2s_nstatus_pin RW 0x1	f2s_initdone_oe RW 0x1	f2s_usermode RW 0x1	f2s_early_usermode RW 0x1	f2s_crc_error RW 0x1
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dma_config 0x90	Reserved															clearFifo 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							dmareq_enable 0x0	dmareq_level 0x0							

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
imgcfg_fifo_status 0x94	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						FifoEmpty 0x1	FifoFull 0x0	FifoLevel 0x0							

dclkcnt

Used to give software control in enabling DCLK at any time.

SW will need control of the DCLK in specific configuration and partial reconfiguration initialization steps to send spurious DCLKs required by the CB. SW takes ownership for DCLK during normal configuration, partial reconfiguration, error scenerio handshakes including SEU CRC error during partial reconfiguration, SW early abort of partial reconfiguration, and initializatin phase DCLK driving.

During initialization phase, a configuration image loaded into the FPGA can request that DCLK be used as the initialization phase clock instead of the default internal oscillator or optionally the CLKUSR pin. In the case that DCLK is requested, the DCLKCNT register is used by software to control DCLK during the initialization phase.

Software should poll the DCLKSTAT.DCNTDONE write one to clear register to be set when the correct number of DCLKs have completed. Software should clear DCLKSTAT.DCNTDONE before writing to the DCLKCNT register again.

This field only affects the FPGA if CTRL.EN is 1.

Module Instance	Base Address	Register Address
i_fpga_mgr_fpgamgrregs	0xFFD03000	0xFFD03008

Offset: 0x8

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RW 0x0															

dclkcnt Fields

Bit	Name	Description	Access	Reset
31:0	cnt	Controls DCLK counter. Software writes a non-zero value into CNT and the FPGA Manager generates the specified number of DCLK pulses and decrements COUNT. This register will read back the original value written by software. Software can write CNT at any time.	RW	0x0

dclkstat

This write one to clear register indicates that the DCLKCNT has counted down to zero. The DCLKCNT is used by software to drive spurious DCLKs to the FPGA. Software will poll this bit after writing DCLKCNT to know when all of the DCLKs have been sent.

Module Instance	Base Address	Register Address
i_fpga_mgr_fpgamgrregs	0xFFD03000	0xFFD0300C

Offset: 0xC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															dcntdone RW 0x0

dclkstat Fields

Bit	Name	Description	Access	Reset
0	dcntdone	This bit is write one to clear. This bit gets set after the DCLKCNT has counted down to zero (transition from 1 to 0). Value: Description 0: notdone 1: done	RW	0x0

gpo

Provides a low-latency, low-performance, and simple way to drive general-purpose signals to the FPGA fabric.

Module Instance	Base Address	Register Address
i_fpga_mgr_fpgamgrregs	0xFFD03000	0xFFD03010

Offset: 0x10

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
value RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
value RW 0x0															

gpo Fields

Bit	Name	Description	Access	Reset
31:0	value	Drives s2f_gp[31:0] with specified value. When read, returns the current value being driven to the FPGA fabric.	RW	0x0

gpi

Provides a low-latency, low-performance, and simple way to read general-purpose signals driven from the FPGA fabric.

Module Instance	Base Address	Register Address
i_fpga_mgr_fpgamgrregs	0xFFD03000	0xFFD03014

Offset: 0x14

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
value RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
value RO 0x0															

gpi Fields

Bit	Name	Description	Access	Reset
31:0	value	The value being driven from the FPGA fabric on f2s_gp[31:0]. If the FPGA is not in User Mode, the value of this field is undefined.	RO	0x0

misci

Provides a low-latency, low-performance, and simple way to read specific handshaking signals driven from the FPGA fabric.

Module Instance	Base Address	Register Address
i_fpga_mgr_fpgamgrregs	0xFFD03000	0xFFD03018

Offset: 0x18

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														bootFPGArdy RO 0x0	bootFPGAfail RO 0x0

misci Fields

Bit	Name	Description	Access	Reset
1	bootFPGArdy	The value of the f2s_boot_from_fpga_ready signal from the FPGA fabric. If the FPGA is not in User Mode, the value of this field is undefined. 1 = FPGA fabric is ready to accept AXI master requests from the HPS2FPGA bridge. 0 = FPGA fabric is not ready (probably still processing a reset).	RO	0x0
0	bootFPGAfail	The value of the f2s_boot_from_fpga_on_failure signal from the FPGA fabric. If the FPGA is not in User Mode, the value of this field is undefined. 1 = Boot ROM will boot from FPGA if boot from normal boot device fails. 0 = Boot ROM will not boot from FPGA if boot from normal boot device fails.	RO	0x0

emr_data0

bits [31:0] of EMR register

Module Instance	Base Address	Register Address
i_fpga_mgr_fpgamgrregs	0xFFD03000	0xFFD03030

Offset: 0x30

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
value RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
value RO 0x0															

emr_data0 Fields

Bit	Name	Description	Access	Reset
31:0	value	EMR register bits	RO	0x0

emr_data1

bits [63:32] of EMR register

Module Instance	Base Address	Register Address
i_fpga_mgr_fpgamgrregs	0xFFD03000	0xFFD03034

Offset: 0x34

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
value RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
value RO 0x0															

emr_data1 Fields

Bit	Name	Description	Access	Reset
31:0	value	EMR register bits	RO	0x0

emr_data2

bits [95:64] of EMR register

Module Instance	Base Address	Register Address
i_fpga_mgr_fpgamgrregs	0xFFD03000	0xFFD03038

Offset: 0x38

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
value RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
value RO 0x0															

emr_data2 Fields

Bit	Name	Description	Access	Reset
31:0	value	EMR register bits	RO	0x0

emr_data3

bits [119:96] of EMR register

Module Instance	Base Address	Register Address
i_fpga_mgr_fpgamgrregs	0xFFD03000	0xFFD0303C

Offset: 0x3C

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
value RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
value RO 0x0															

emr_data3 Fields

Bit	Name	Description	Access	Reset
31:0	value	EMR register bits	RO	0x0

emr_data4

bits [159:128] of EMR register

Module Instance	Base Address	Register Address
i_fpga_mgr_fpgamgrregs	0xFFD03000	0xFFD03040

Offset: 0x40

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
value RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
value RO 0x0															

emr_data4 Fields

Bit	Name	Description	Access	Reset
31:0	value	EMR register bits	RO	0x0

emr_data5

bits [171:160] of EMR register

Module Instance	Base Address	Register Address
i_fpga_mgr_fpgamgrregs	0xFFD03000	0xFFD03044

Offset: 0x44

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
value RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
value RO 0x0															

emr_data5 Fields

Bit	Name	Description	Access	Reset
31:0	value	EMR register bits	RO	0x0

emr_valid

Module Instance	Base Address	Register Address
i_fpga_mgr_fpgamgrregs	0xFFD03000	0xFFD03048

Offset: 0x48

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															vld 0x0

emr_valid Fields

Bit	Name	Description	Access	Reset
0	vld		RW	0x0

emr_en

Module Instance	Base Address	Register Address
i_fpga_mgr_fpgamgrregs	0xFFD03000	0xFFD0304C

Offset: 0x4C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															en RW 0x0

emr_en Fields

Bit	Name	Description	Access	Reset
0	en	Enable HPS EMR extraction from CSS.	RW	0x0

jtag_config

This register is used to configure the JTAG master interface.

It is recommended that software write this register before initiating a transfer.
If the software writes to this register while an active session is in progress (as indicated by jtag_status.SessionStatus), the expected behavior is "undefined".

Module Instance	Base Address	Register Address
i_fpga_mgr_fpgamgrregs	0xFFD03000	0xFFD03050

Offset: 0x50

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
txSize RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
tckRatio RW 0x14								Reserved			trstEn RW 0x0	Reserved	loopBackEn RW 0x0	jtagPortEn RW 0x0	JtagHostEn RW 0x0

jtag_config Fields

Bit	Name	Description	Access	Reset
31:16	txSize	Defines the number of bits to be transmitted. Once the software kicks of the transfer via Start_ Transfer bits, the hardware will stop the transfer, when either of the below conditions are reached. a) Completed a transfer of programmed Number of TX Bits +1. b) TxFifo became empty (under-run of TxFifo). c) A Stop Transfer request received from software. So for a successful transfer of fixed number of bits, software has to make sure continuous flow of data. The exact number of bits to be transferred in the current session is 1+ the value in this register field. For example 0 -> 1 bit to be transmitted once start session is triggered. 1-> 2 bits to be transmitted.once start session is triggered.	RW	0x0
15:8	tckRatio	Ratio of TCK division. The FPGA manager clock is 100MHz. value of 4 provides a 25MHz TCK value of 20 (0x14) provides a 5MHz TCK. Maximum supported TCK frequency is 25MHz. So writing a value less than 4 to this field will cause unexpected behavior.	RW	0x14

Bit	Name	Description	Access	Reset
4	trstEn	Set this bit and then writing at least 1 data in jtag_data_w will cause a JTAG reset to happen. Please note that writing TRSTEN while a data transmission is undergoing could cause undesired effects, so it is recommended that software poll the SESSIONSTATUS bit to make sure there is no existing transfers before writing TRSTEN. Software should manually write 0 to this bit after the completion of the reset and after the SESSION-STATUS is inactive.	RW	0x0
2	loopBackEn	Enables the internal loopback mode. A typical scenario will be to set JtagHostEn=0, JtagPortEn=0, and then initiate transmits by software writes to TXFifo. All transfers should receive back on RxFifo. Also this should not affect anything external, since the Jtag ports will be gated off. **Value** **Description** 0 disable 1 enable	RW	0x0
1	jtagPortEn	This bit field gates off TDI/TMS/TCK driven to the FPGA CSS interface. This allows the software to take over JTAG but keep them tied low. If this bit is 0 the internal core logic will still be active and all the status will be updated as in regular operation. **Value** **Description** 0 disable 1 enable	RW	0x0

Bit	Name	Description	Access	Reset
0	JtagHostEn	This bit field drives the enable signal to the FPGA CSS. Please note that this enable should be driven long before you start any JTAG transactions. Value: Description 0: disable 1: enable	RW	0x0

jtag_status

status of the currently ongoing JTAG transmit or receive

Module Instance	Base Address	Register Address
i_fpga_mgr_fpgamgrregs	0xFFD03000	0xFFD03054

Offset: 0x54

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
txDoneSize 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Session-Status 0x0	Reserved			rxFifoFull 0x0	rxFifoEmpty 0x1	txFifoFull 0x0	txFifoEmpty 0x1	rxFifoLevel 0x0				txFifoLevel 0x0			

jtag_status Fields

Bit	Name	Description	Access	Reset
31:16	txDoneSize	Total Number of Successful bits transferred in the current session. The exact number of bits transferred in the current session is 1+ the value in this register. For example 0 -> 1 bit completed transmit or in the process of transmitting. 1-> 2 bits completed transmit or in the process of transmitting.	RO	0x0
15	SessionStatus	A read of 1 indicates that there is an ongoing transfer session. A read of 0 indicates that there is no ongoing transfer session. **Value** — **Description** 0 — disable 1 — enable	RO	0x0
11	rxFifoFull	Read 1 -> Rx Fifo Full 0 -> Rx Fifo NOT full	RO	0x0
10	rxFifoEmpty	Read 1 -> Rx Fifo Empty 0 -> Rx Fifo NOT Empty	RO	0x1
9	txFifoFull	Read 1 -> Tx Fifo Full 0 -> Tx Fifo NOT full	RO	0x0
8	txFifoEmpty	Read 1 -> Tx Fifo Empty 0 -> Tx Fifo NOT Empty	RO	0x1
7:4	rxFifoLevel	Number of Words remaining in the Rx Fifo. Maximum value is 0x8	RO	0x0
3:0	txFifoLevel	Number of words remaining in Tx Fifo. Maximum value is 0x8	RO	0x0

jtag_kick

Jtag Master Control Triggers. Each of this bit field triggers a specific hardware operation.

Module Instance	Base Address	Register Address
i_fpga_mgr_fpgamgrregs	0xFFD03000	0xFFD03058

Offset: 0x58

Access: WO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												clearRxFifo 0x0	clearTxFifo 0x0	stopSession 0x0	startSession 0x0

jtag_kick Fields

Bit	Name	Description	Access	Reset
3	clearRxFifo	A write 1 to this bit field will empty the RxFifo. A read will always return 0.	WO	0x0
2	clearTxFifo	A write 1 to this bit field will empty the TxFifo. A read will always return 0.	WO	0x0

Bit	Name	Description	Access	Reset
1	stopSession	A write 1 to this bit field will kick off stop of an ongoing session of TX and RX. Please note that there should be alteast 1 word available in the Tx Fifo for this kick off to be successful. The status of this kick off can be read from status register SessionStatus field. stopSession has priority over startSession.	WO	0x0
0	startSession	A write 1 to this bit field will kick off a session of TX and RX. Please note that there should be alteast 1 word available in the Tx Fifo for this kick off to be successful. The status of this kick off can be read from status register SessionStatus field. stopSession has priority over startSession.	WO	0x0

jtag_data_w

A write to this field initiates a write to the TxFifo with the value provided. Please note that this is always a 32bit write, and lower 16 of these bits will be transferred over TDI and upper 16 of these bits will be transferred over TMS once the transfer is initiated. A write to this fifo when full will just be ignored. Also it is allowed to keep writing this fifo without actually intiating a transfer. A typical scenario where this is applicable is if the software want to keep a fixed number of bits ready for JTAG interface before kicking it off.

Both the fifos are 8 words deep. So you can have up to 128 bits buffered in the FIFOs. If you have more than 128 bits to be transferred, and want continuous transfer software should make sure proper data flow to avoid a Tx-Fifo under-run or Rx-Fifo over-run. A Rx-Fifo over-run will cause silent data loss, and a Tx-Fifo under-run will stop the transfer, and will stop the TCK toggles, till another transfer is initiated by software.
At 5MHz jtag clock 16x100/5 = 320 cycles of FPGA manager clock to transfer 1 word of data
At 25Mhz jtag clock 16x100/25 = 64 cycles of FPGA manager clock to transfer 1 word of data.

Module Instance	Base Address	Register Address
i_fpga_mgr_fpgamgrregs	0xFFD03000	0xFFD03060

Offset: 0x60

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
tmsData 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
tdiData 0x0															

jtag_data_w Fields

Bit	Name	Description	Access	Reset
31:16	tmsData	Data to be transferred over TMS.	WO	0x0
15:0	tdiData		WO	0x0

jtag_data_r

A read to this field initiates a read to the RxFifo. Please note that the read always returns a 16 bit value. If the received number of bits are not aligned with 16, lower most n bits should be ignored by software. For example in the case of 6 bits received, rdata[15:10] will contain received bits and rdata[9:0] should be ignored. A read of the fifo while empty is not defined.

Both the fifos are 8 words deep. So you can have up to 128 bits buffered in the FIFOs. If you have more than 128 bits to be transferred, and want continuous transfer software should make sure proper data flow to avoid a Tx-Fifo under-run or Rx-Fifo over-run. A Rx-Fifo over-run will cause silent data loss, and a Tx-Fifo under-run will stop the transfer, and will stop the TCK toggles, till another transfer is initiated by software.
At 5MHz jtag clock 16x100/5 = 320 cycles of FPGA manager clock to transfer 1 word of data
At 25Mhz jtag clock 16x100/25 = 64 cycles of FPGA manager clock to transfer 1 word of data.

Module Instance	Base Address	Register Address
i_fpga_mgr_fpgamgrregs	0xFFD03000	0xFFD03064

Offset: 0x64

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
tdiData 0x0															

jtag_data_r Fields

Bit	Name	Description	Access	Reset
31:16	Reserved		RO	0x0
15:0	tdiData		RO	0x0

imgcfg_ctrl_00

Module Instance	Base Address	Register Address
i_fpga_mgr_fpgamgrregs	0xFFD03000	0xFFD03070

Offset: 0x70

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							s2f_condone_oe RW 0x0	Reserved							s2f_nstatus_oe RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							s2f_nconfig RW 0x1	Reserved					s2f_nenable_condone RW 0x1	s2f_nenable_nstatus RW 0x1	s2f_nenable_nconfig RW 0x1

imgcfg_ctrl_00 Fields

Bit	Name	Description	Access	Reset
24	s2f_condone_oe	This bit has an effect on CSS/PIN only if curresponding override is enabled. 1: Drive CONF_DONE pin to 0. 0: Disable CONF_DONE pin drive from HPS, and allow default pull-up to take over. HPS can drive CONF_DONE pin 0 to delay the FPGA from entering user mode.	RW	0x0
16	s2f_nstatus_oe	This bit has an effect on CSS only if curresponding override is enabled. 1: Drive nSTATUS pin to 0. 0: Disable nSTATUS pin drive from HPS, and allow default pull-up to take over. Driving this pin has no effect on CSS once the FPGA is in user mode. HPS can drive this pin 0 to delay the initialization phase of CSS. During configuation phase, HPS can drive this pin 0 to end the current configuration and initialize a reconfiguration.	RW	0x0

Bit	Name	Description	Access	Reset
8	s2f_nconfig	This bit has an effect on CSS only if curresponding override is enabled. 1: Drive nCONFIG input to CSS 1. 0: Drive nCONFIG input to CSS 0 The nCONFIG input is used to put the FPGA into its reset phase. If the FPGA was configured, its operation stops and it will have to be configured again to start operation.	RW	0x1
2	s2f_nenable_condone	HPS override Enable for CONF_DONE to PIN 1: override is disabled 0: override is enabled	RW	0x1
1	s2f_nenable_nstatus	HPS override Enable for nSTATUS to PIN 1: override is disabled 0: override is enabled	RW	0x1
0	s2f_nenable_nconfig	HPS override Enable for nCONFIG to CSS 1: override is disabled 0: override is enabled	RW	0x1

imgcfg_ctrl_01

Module Instance	Base Address	Register Address
i_fpga_mgr_fpgamgrregs	0xFFD03000	0xFFD03074

Offset: 0x74

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							s2f_nce RW 0x1	Reserved							s2f_pr_request RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															s2f_nenable_config RW 0x1

imgcfg_ctrl_01 Fields

Bit	Name	Description	Access	Reset
24	s2f_nce	This bit carries the override value of nCE to CSS.	RW	0x1
16	s2f_pr_request	This bit carries the override value of PR_REQUEST to CSS.	RW	0x0
0	s2f_nenable_config	HPS override Enable for DATA, DCLK, NCE and PR_REQUEST to CSS 1: override is disabled 0: override is enabled	RW	0x1

imgcfg_ctrl_02

Allows HPS to control FPGA configuration.

The NCONFIGPULL, NSTATUSPULL, and CONFDONEPULL fields drive signals to the FPGA Control Block that are logically ORed into their respective pins. These signals are always driven independent of the value of EN. The polarity of the NCONFIGPULL, NSTATUSPULL, and CONFDONEPULL fields is inverted relative to their associated pins.

The MSEL (external pins), CDRATIO and CFGWDTH signals determine the mode of operation for Normal Configuration. For Partial Reconfiguration, CDRATIO is used to set the appropriate clock to data ratio, and CFGWDTH should always be set to 16-bit Passive Parallel.

AXICFGEN is used to enable transfer of configuration data by enabling or disabling DCLK during data transfers.

Module Instance	Base Address	Register Address
i_fpga_mgr_fpgamgrregs	0xFFD03000	0xFFD03078

Offset: 0x78

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							cfgwidth RW 0x0	Reserved						cdratio RW 0x0	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							en_cfg_data RW 0x0	Reserved							en_cfg_ctrl 0x0

imgcfg_ctrl_02 Fields

Bit	Name	Description	Access	Reset
24	cfgwidth	This field determines the Configuration Passive Parallel data bus width when HPS configures the FPGA. Only 32-bit Passive Parallel or 16-bit Passive Parallel are supported. When HPS does Normal Configuration, configuration should use 32-bit Passive Parallel Mode. The external pins MSEL must be set appropriately for the configuration selected. For Partial Reconfiguration, 16-bit Passive Parallel must be used. **Value** **Description** 0 PPx16 1 PPx32	RW	0x0

Bit	Name	Description	Access	Reset
17:16	cdratio	This field controls the Clock to Data Ratio (CDRATIO) for Normal Configuration and Partial Reconfiguration data transfer from the AXI Slave to the FPGA. For Normal Configuration, the value in this field must be set to be consistent to the implied CD ratio of the MSEL setting. For Partial Reconfiguration, the value in this field must be set to the same clock to data ratio in the options bits in the Normal Configuration file. **Value** — **Description** 0 — x1 1 — x2 2 — x4 3 — x8	RW	0x0
8	en_cfg_data	this is an unused software bit	RW	0x0
0	en_cfg_ctrl	If this bit is not enabled, the s2f_dclk as well as s2f_data will be always driven 0. This is to provide a mechanism by which HPS can take over the DCLK/DATA by first setting the nenable_dclk even while s2f_dclk and s2f_data from HPS is silent.	RW	0x0

imgcfg_stat

This is the unmasked status.
Value of corresponding inputs from CSS or PINs, without considering the intr_mask or intr_polarity.

Module Instance	Base Address	Register Address
i_fpga_mgr_fpgamgrregs	0xFFD03000	0xFFD03080

Offset: 0x80

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		emr RO 0x0	jtagm RO 0x0	Reserved		imgcfg_FifoFull RO 0x0	imgcfg_FifoEmpty RO 0x1	Reserved					f2s_msel2 RO 0x0	f2s_msel1 RO 0x0	f2s_msel0 RO 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		f2s_nceo_oe RO 0x0	f2s_nconfig_pin RO 0x0	f2s_pr_error RO 0x0	f2s_pr_done RO 0x0	f2s_pr_ready RO 0x0	f2s_cvp_conf_done RO 0x0	f2s_condone_oe RO 0x0	f2s_condone_pin RO 0x0	f2s_nstatus_oe RO 0x0	f2s_nstatus_pin RO 0x0	f2s_initdone_oe RO 0x0	f2s_usermode RO 0x0	f2s_early_usermode RO 0x0	f2s_crc_error RO 0x0

imgcfg_stat Fields

Bit	Name	Description	Access	Reset
29	emr	EMR valid bit	RO	0x0
28	jtagm	JTAG Master Session Status	RO	0x0
25	imgcfg_FifoFull	FIfoFull Status of FPGA image configuration FIFO	RO	0x0
24	imgcfg_FifoEmpty	FIfoEmpty Status of FPGA image configuration FIFO	RO	0x1
18	f2s_msel2	This read-only field allows software to observe the MSEL inputs from the device pins. The MSEL pins define the FPGA configuration mode. Please refer to CSS functional specifications for the exact definisions of MSEL encoding. In Baum only 3 of these are used. Other bits will always read 0.	RO	0x0

Bit	Name	Description	Access	Reset
17	f2s_msel1	This read-only field allows software to observe the MSEL inputs from the device pins. The MSEL pins define the FPGA configuration mode. Please refer to CSS functional specifications for the exact definisions of MSEL encoding. In Baum only 3 of these are used. Other bits will always read 0.	RO	0x0
16	f2s_msel0	This read-only field allows software to observe the MSEL inputs from the device pins. The MSEL pins define the FPGA configuration mode. Please refer to CSS functional specifications for the exact definisions of MSEL encoding. In Baum only 3 of these are used. Other bits will always read 0.	RO	0x0
13	f2s_nceo_oe	Chip select output enable driven from Control Subsystem.	RO	0x0
12	f2s_nconfig_pin	The value of this pin is used for monitoring purposes only in the FPGA manager.	RO	0x0
11	f2s_pr_error	Partial reconfiguration error	RO	0x0
10	f2s_pr_done	Partial reconfiguration done status	RO	0x0
9	f2s_pr_ready	Partial reconfiguration ready	RO	0x0
8	f2s_cvp_conf_done	Configuration via PCIe done indicator.	RO	0x0
7	f2s_condone_oe	This is the driven output enable of condone from the Control Subsystem (CSS). This signal can be used by software to determine the true status of the CSS.	RO	0x0
6	f2s_condone_pin	Sampled pin value for monitoring purposes only in FPGA manager. Please note that this value can be overriden by external devices.	RO	0x0

Bit	Name	Description	Access	Reset
5	f2s_nstatus_oe	This is the driven output enable of nstatus from the Control Subsystem (CSS). This signal can be used by software to determine true status of the CSS.	RO	0x0
4	f2s_nstatus_pin	This status bit holds the signal value of nstatus which the Control Subsystem (CSS) sees when override is enabled, irrespective of what the pin value is. When this signal is 1, it doesn't affect the pin value. When this signal is 1, it pulls down the pin value (irrespective override enable).	RO	0x0
3	f2s_initdone_oe	This bit indicates the true status of the Control Subsystem.	RO	0x0
2	f2s_usermode	User mode status. Asserted only when FPGA has finally entered user mode. The FPGA manager receives this signal after a glitch filter external to FPGA manager. This glitch filtered signal is named fpga_config_complete.	RO	0x0
1	f2s_early_usermode	Status of early user mode signal from the Control Subsystem (CSS). Used by software to determine status of when HPS is configuring the shared IOs via sending POF to CSS.	RO	0x0
0	f2s_crc_error	CRC error indicator.	RO	0x0

intr_masked_status

When you read this register you read the active high pending interrupt status of corresponding bit.
This value is after the masking specified by intr_mask and after the polarity conversion as specified in intr_polarity

Module Instance	Base Address	Register Address
i_fpga_mgr_fpgamgrregs	0xFFD03000	0xFFD03084

Offset: 0x84

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		emr RW 0x0	jtagm RW 0x0	Reserved		imgcfg_FifoFull RW 0x0	imgcfg_FifoEmpty RW 0x0	Reserved					f2s_msel2 RW 0x0	f2s_msel1 RW 0x0	f2s_msel0 RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		f2s_nceo_oe RW 0x0	f2s_nconfig_pin RW 0x0	f2s_pr_error RW 0x0	f2s_pr_done RW 0x0	f2s_pr_ready RW 0x0	f2s_cvp_conf_done RW 0x0	f2s_condone_oe RW 0x0	f2s_condone_pin RW 0x0	f2s_nstatus_oe RW 0x0	f2s_nstatus_pin RW 0x0	f2s_initdone_oe RW 0x0	f2s_usermode RW 0x0	f2s_early_usermode RW 0x0	f2s_crc_error RW 0x0

intr_masked_status Fields

Bit	Name	Description	Access	Reset
29	emr	EMR valid bit	RW	0x0
28	jtagm	JTAG Master Session Status	RW	0x0
25	imgcfg_FifoFull	FIfoFull Status of FPGA image configuration FIFO	RW	0x0
24	imgcfg_FifoEmpty	FIfoEmpty Status of FPGA image configuration FIFO	RW	0x0
18	f2s_msel2		RW	0x0
17	f2s_msel1		RW	0x0
16	f2s_msel0		RW	0x0
13	f2s_nceo_oe		RW	0x0
12	f2s_nconfig_pin		RW	0x0
11	f2s_pr_error		RW	0x0
10	f2s_pr_done		RW	0x0
9	f2s_pr_ready		RW	0x0
8	f2s_cvp_conf_done		RW	0x0

Bit	Name	Description	Access	Reset
7	f2s_condone_oe		RW	0x0
6	f2s_condone_pin		RW	0x0
5	f2s_nstatus_oe		RW	0x0
4	f2s_nstatus_pin		RW	0x0
3	f2s_initdone_oe		RW	0x0
2	f2s_usermode		RW	0x0
1	f2s_early_usermode		RW	0x0
0	f2s_crc_error		RW	0x0

intr_mask

Mask for interrupts. A value of 1 in a particular bit will cause the specific interrupt to be masked.

Module Instance	Base Address	Register Address
i_fpga_mgr_fpgamgrregs	0xFFD03000	0xFFD03088

Offset: 0x88

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		emr RW 0x1	jtagm RW 0x1	Reserved		imgcfg_FifoFull RW 0x1	imgcfg_FifoEmpty RW 0x1	Reserved					f2s_msel2 RW 0x1	f2s_msel1 RW 0x1	f2s_msel0 RW 0x1
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		f2s_nceo_oe RW 0x1	f2s_nconfig_pin RW 0x1	f2s_pr_error RW 0x1	f2s_pr_done RW 0x1	f2s_pr_ready RW 0x1	f2s_cvp_conf_done RW 0x1	f2s_condone_oe RW 0x1	f2s_condone_pin RW 0x1	f2s_nstatus_oe RW 0x1	f2s_nstatus_pin RW 0x1	f2s_initdone_oe RW 0x1	f2s_usermode RW 0x1	f2s_early_usermode RW 0x1	f2s_crc_error RW 0x1

intr_mask Fields

Bit	Name	Description	Access	Reset
29	emr	EMR valid bit	RW	0x1
28	jtagm	JTAG Master Session Status	RW	0x1
25	imgcfg_FifoFull	FIfoFull Status of FPGA image configuration FIFO	RW	0x1
24	imgcfg_FifoEmpty	FIfoEmpty Status of FPGA image configuration FIFO	RW	0x1
18	f2s_msel2		RW	0x1
17	f2s_msel1		RW	0x1
16	f2s_msel0		RW	0x1
13	f2s_nceo_oe		RW	0x1
12	f2s_nconfig_pin		RW	0x1
11	f2s_pr_error		RW	0x1
10	f2s_pr_done		RW	0x1
9	f2s_pr_ready		RW	0x1
8	f2s_cvp_conf_done		RW	0x1
7	f2s_condone_oe		RW	0x1
6	f2s_condone_pin		RW	0x1
5	f2s_nstatus_oe		RW	0x1
4	f2s_nstatus_pin		RW	0x1
3	f2s_initdone_oe		RW	0x1
2	f2s_usermode		RW	0x1
1	f2s_early_usermode		RW	0x1
0	f2s_crc_error		RW	0x1

intr_polarity

Active Level of the signal to generate interrupt.

0 :Active LOW. An interrupt will be generated when that particular bit/signal is LOW
1: Active HIGH. An interrupt will be generated when that particular bit/signal is HIGH

Module Instance	Base Address	Register Address
i_fpga_mgr_fpgamgrregs	0xFFD03000	0xFFD0308C

Offset: 0x8C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		emr RW 0x1	jtagm RW 0x1	Reserved		imgcfg_FifoFull RW 0x1	imgcfg_FifoEmpty RW 0x1	Reserved					f2s_msel2 RW 0x1	f2s_msel1 RW 0x1	f2s_msel0 RW 0x1
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		f2s_nceo_oe RW 0x1	f2s_nconfig_pin RW 0x1	f2s_pr_error RW 0x1	f2s_pr_done RW 0x1	f2s_pr_ready RW 0x1	f2s_cvp_conf_done RW 0x1	f2s_condone_oe RW 0x1	f2s_condone_pin RW 0x1	f2s_nstatus_oe RW 0x1	f2s_nstatus_pin RW 0x1	f2s_initdone_oe RW 0x1	f2s_usermode RW 0x1	f2s_early_usermode RW 0x1	f2s_crc_error RW 0x1

intr_polarity Fields

Bit	Name	Description	Access	Reset
29	emr	EMR valid bit	RW	0x1
28	jtagm	JTAG Master Session Status	RW	0x1
25	imgcfg_FifoFull	FIfoFull Status of FPGA image configuration FIFO	RW	0x1
24	imgcfg_FifoEmpty	FIfoEmpty Status of FPGA image configuration FIFO	RW	0x1
18	f2s_msel2		RW	0x1
17	f2s_msel1		RW	0x1
16	f2s_msel0		RW	0x1
13	f2s_nceo_oe		RW	0x1
12	f2s_nconfig_pin		RW	0x1
11	f2s_pr_error		RW	0x1

Bit	Name	Description	Access	Reset
10	f2s_pr_done		RW	0x1
9	f2s_pr_ready		RW	0x1
8	f2s_cvp_conf_done		RW	0x1
7	f2s_condone_oe		RW	0x1
6	f2s_condone_pin		RW	0x1
5	f2s_nstatus_oe		RW	0x1
4	f2s_nstatus_pin		RW	0x1
3	f2s_initdone_oe		RW	0x1
2	f2s_usermode		RW	0x1
1	f2s_early_usermode		RW	0x1
0	f2s_crc_error		RW	0x1

dma_config

Consist of control bit and status information.

Module Instance	Base Address	Register Address
i_fpga_mgr_fpgamgrregs	0xFFD03000	0xFFD03090

Offset: 0x90

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															clearFifo 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							dmareq_enable 0x0	dmareq_level 0x0							

dma_config Fields

Bit	Name	Description	Access	Reset
16	clearFifo	A write 1 to this bit field will empty the TxFifo. A read will always return 0.	RW	0x0
8	dmareq_enable	Writing 1 will enable DMA request handshake from FPGA manager. Writing 0 will disable DMA request handshake from FPGA manager. **Value** **Description** 0 DMA request handshake disable. 1 DMA request handshake enable	RW	0x0
7:0	dmareq_level	DMA request threshold level	RW	0x0

imgcfg_fifo_status

Consist of control bit and status information.

Module Instance	Base Address	Register Address
i_fpga_mgr_fpgamgrregs	0xFFD03000	0xFFD03094

Offset: 0x94

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						FifoEmpty 0x1	FifoFull 0x0	FifoLevel 0x0							

imgcfg_fifo_status Fields

Bit	Name	Description	Access	Reset
9	FifoEmpty	Read 1 -> Fifo Empty 0 -> Fifo NOT Empty	RO	0x1
8	FifoFull	Read 1 -> Fifo Full 0 -> Fifo NOT full	RO	0x0
7:0	FifoLevel	Number of words remaining in FPGA Image Configuration Fifo. Maximum value is 0x64	RO	0x0

Document Revision History

Table 4-2: Document Revision History

Date	Version	Changes
October 2016	2016.10.28	Maintenance release
May 2016	2016.05.03	Maintenance release
November 2015	2015.11.02	• Provided more information for the configuration schemes for the dedicated pins. • Added missing address maps and register definitions.
May 2015	2015.05.04	Maintenance release
December 2014	2014.12.15	Maintenance release
August 2014	2014.08.18	Initial release

System Manager 5

2016.10.28

a10_5v4 Subscribe Send Feedback

The system manager in the hard processor system (HPS) contains memory-mapped control and status registers (CSRs) and logic to control system level functions as well as other modules in the HPS.

The system manager connects to the following modules in the HPS:

- Direct memory access (DMA) controller
- Ethernet media access controllers (EMAC0, EMAC1, and EMAC2)
- Error Checking and Correction Controller (ECC) for RAMs
- Microprocessor unit (MPU) subsystem
- NAND flash controller
- Secure Digital/MultiMediaCard (SD/MMC) controller
- Quad serial peripheral interface (SPI) flash controller
- USB 2.0 On-The-Go (OTG) controllers (USB0 and USB1)
- Watchdog timers

Features of the System Manager

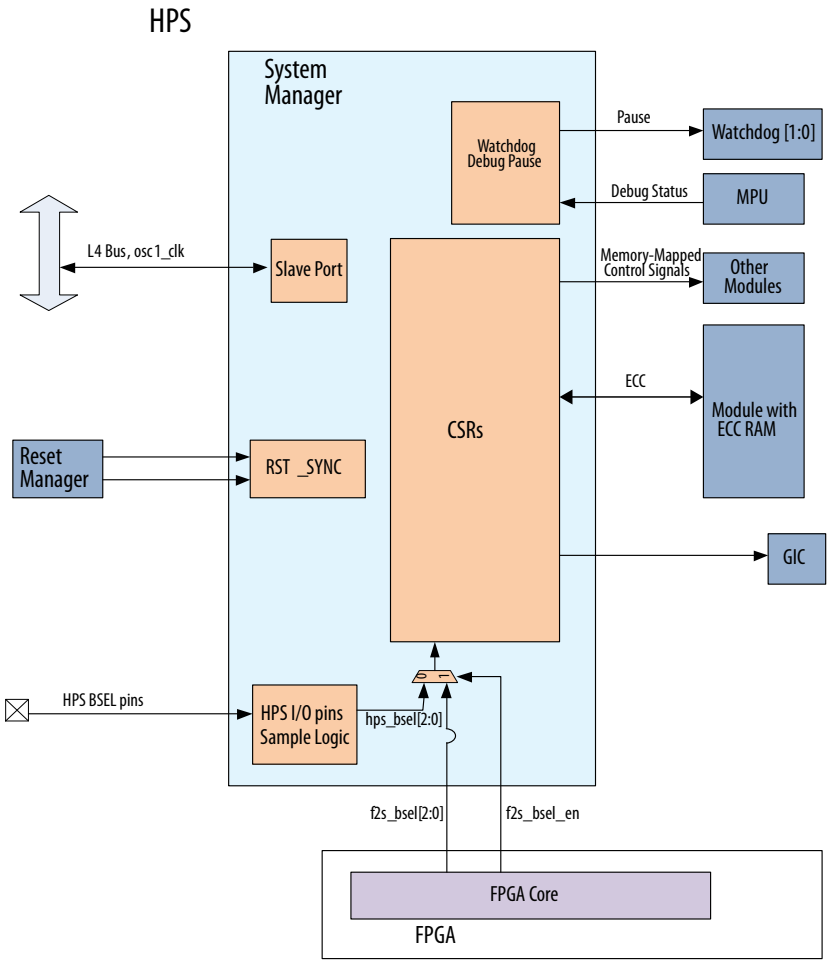
Software accesses the CSRs in the system manager to control and monitor various functions in other HPS modules that require external control signals. The system manager connects to these modules to perform the following functions:

- Sends pause signals to pause the watchdog timers when the processors in the MPU subsystem are in debug mode
- Selects the EMAC system interconnect master access options and other EMAC clock and interface options.
- Selects the SD/MMC controller clock options and system interconnect master access options.
- Selects the NAND flash controller bootstrap options and system interconnect master access option.
- Selects USB controller system interconnect master access option.
- Provides control over the DMA security settings when the HPS exits from reset.
- Provides boot source information that can be read during the boot process.
- Provides the capability to enable or disable an interface to the FPGA.
- Provides combined ECC status and interrupts from other HPS modules with ECC-protected RAM.
- Routes parity failure interrupts from the L1 caches to the Global Interrupt Controller.
- Provides the capability to inject errors during testing in the MPU L2 ECC-protected RAM.

System Manager Block Diagram and System Integration

The system manager connects to the level 4 (L4) bus through a slave interface. The CSRs connect to signals in the FPGA and other HPS modules.

Figure 5-1: System Manager Block Diagram



The system manager consists of the following:

- CSRs—Provide memory-mapped access to control signals and status for the following HPS modules:
  - EMACs
  - Debug core
  - SD/MMC controller
  - NAND controller
  - USB controllers
  - DMA controller
  - System interconnect
  - ECC memory interfaces for the following peripherals:
    - USB controllers
    - SD/MMC controller
    - Ethernet MACs
    - DMA controller
    - NAND flash controller
    - On-chip RAM
- Slave port interface—provides access to system manager CSRs for connected masters.
- Watchdog debug pause—accepts the debug mode status from the MPU subsystem and pauses the L4 watchdog timers.
- Reset Manager— system manager receives the reset signals from reset manager.

Functional Description of the System Manager

The system manager serves the following purposes:

- Provides software access to boot configuration and system information
- Provides software access to control and status signals in other HPS modules
- Provides combined ECC status and interrupt from other HPS modules with ECC-protected RAM
- Enables and disables HPS peripheral interfaces to the FPGA
- Provides eight registers that software can use to pass information between boot stages

Boot Configuration and System Information

The system manager provides boot configuration information through the `bootinfo` register. Sampled value of the HPS boot select (`BSEL`) pins are available to the Boot ROM software.

The boot source is determined by a combination of the `BSEL` pins and a fuse bit that can bypass the `BSEL` pins.

Related Information

Booting and Configuration on page 30-1

Additional Module Control

Each module in the HPS has its own CSRs, providing access to the internal state of the module. The system manager provides registers for additional module control and monitoring. To fully control each module,

you must program both the peripheral's CSR and its corresponding CSR in the System Manager. This section describes how to configure the system manager CS for each module.

DMA Controller

The security state of the DMA controller is controlled by the manager thread security (`mgr_ns`) and interrupt security (`irq_ns`) bits of the DMA register.

The `periph_ns` register bits determine if a peripheral request interface is secure or non-secure.

Note: The `periph_ns` register bits must be configured before the DMA is released from global reset.

Related Information

- **DMA Controller** on page 16-1
- **System Manager Address Map and Register Definitions** on page 5-9

NAND Flash Controller

The bootstrap control register (`nand_bootstrap`) modifies the default behavior of the NAND flash controller after reset. The NAND flash controller samples the bootstrap control register bits when it comes out of reset.

The following `nand_bootstrap` register bits control configuration of the NAND flash controller:

- Bootstrap inhibit initialization bit (`noinit`)—inhibits the NAND flash controller from initializing when coming out of reset, and allows software to program all registers pertaining to device parameters such as page size and width.
- Bootstrap 512-byte device bit (`page512`)—informs the NAND flash controller that a NAND flash device with a 512-byte page size is connected to the system.
- Bootstrap inhibit load block 0 page 0 bit (`noloadb0p0`)—inhibits the NAND flash controller from loading page 0 of block 0 of the NAND flash device during the initialization procedure.
- Bootstrap two row address cycles bit (`tworowaddr`)—informs the NAND flash controller that only two row address cycles are required instead of the default three row address cycles.

You can use the system manager's `nanad_l3master` register to control the following signals:

- ARPROT
- AWPROT
- ARDOMAIN
- AWDOMAIN
- ARCACHE
- AWCACHE

These bits define the cache attributes for the master transactions of the DMA engine in the NAND controller.

Note: Register bits must be accessed only when the master interface is guaranteed to be in an inactive state.

Related Information

- **NAND Flash Controller** on page 13-1
- **System Manager Address Map and Register Definitions** on page 5-9

EMAC

You can program the `emac_global` register to select either `emac_ptp_clk` from the Clock Manager or `f2s_ptp_ref_clk` from the FPGA fabric as the source of the IEEE 1588 reference clock for each EMAC.

You can use the system manager's `l3master` register to control the EMAC's `ARCACHE` and`AWCACHE` signals, by setting or clearing the (`arcache, awcache`) and (`arprot, awprot`) bits. These bits define the cache attributes for the master transactions of the DMA engine in the EMAC controllers.

Note: Register bits must be accessed only when the master interface is guaranteed to be in an inactive state.

The `phy_intf_sel` bit is programmed to select between a GMII (MII), RGMII or RMII PHY interface when the peripheral is released from reset. The `ptp_ref_sel` bit selects if the timestamp reference is internally or externally generated. The `ptp_ref_sel` bit must be set to the correct value before the EMAC core is pulled out of reset.

Note: EMAC0 must be set to internal timestamp.

Related Information

- **Clock Manager** on page 2-1
- **Ethernet Media Access Controller** on page 17-1

USB 2.0 OTG Controller

The `usb*_l3master` registers in the system manager control the `HPROT` and `HAUSER` fields of the USB master port of the USB 2.0 OTG Controller.

Note: Register bits should be accessed only when the master interface is guaranteed to be in an inactive state.

Related Information

USB 2.0 OTG Controller on page 18-1

SD/MMC Controller

The `sdmmc_l3master` register in the system manager controls the `HPROT` and `HAUSER` fields of the SD/MMC master port.

Note: Register bits should be accessed only when the master interface is guaranteed to be in an inactive state.

You can program software to select the clock's phase shift for `cclk_in_drv` and `cclk_in_sample` by setting the drive clock phase shift select (`drvsel`) and sample clock phase shift select (`smplsel`) bits of the `sdmmc` register in the system manager.

Related Information

SD/MMC Controller on page 14-1

Watchdog Timer

The system manager controls the watchdog timer behavior when the CPUs are in debug mode. The system manager sends a pause signal to the watchdog timers depending on the setting of the debug mode bits of

the L4 watchdog debug register (wddbg). Each watchdog timer built into the MPU subsystem is paused when its associated CPU enters debug mode.

Related Information

Watchdog Timer on page 24-1

Boot ROM Code

Registers in the system manager control whether the boot ROM code configures the pin multiplexing for boot pins after a warm reset. Set the warm-reset-configure-pin-multiplex for boot pins bit (warmrstcfg-pinmux) of the boot ROM code register to enable or disable this control.

Note: The boot ROM code always configures the pin multiplexing for boot pins after a cold reset.

Registers in the system manager also control whether the boot ROM code configures the I/O pins used during the boot process after a warm reset. Set the warm reset configure I/Os for boot pins bit (warmrstcfgio) of the ctrl register to enable or disable this control.

Note: By default, the boot ROM code always configures the I/O pins used by boot after a cold reset.

There can be up to four preloader images stored in flash memory. The (initswlastld) register contains the index of the preloader's last image that is loaded in the on-chip RAM.

The boot ROM software state register (bootromswstate) is a 32-bit general-purpose register reserved for the boot ROM.

The following warmram related registers are used to configure the warm reset from on-chip RAM feature and must be written by software prior to the warm reset occurring.

Table 5-1: The warmram Registers

Register	Name	Purpose
enable	Enable	Controls whether the boot ROM attempts to boot from the contents of the on-chip RAM on a warm reset.
datastart	Data start	Contains the byte offset of the warm boot CRC validation region in the on-chip RAM. The offset must be word-aligned to an integer multiple of four.
length	Length	Contains the length in bytes of the region in the on-chip RAM available for warm boot CRC validation.
execution	Execution address	Contains the global address the boot code jumps to in the on-chip RAM if the CRC validation succeeds.
crc	Expected CRC	Contains the expected CRC of the region in the on-chip RAM.

The number of wait states applied to the boot ROM's read operation is determined by the wait state bit (waitstate) of the ctrl register. After the boot process, software might require reading the code in the

boot ROM. If software has changed the clock frequency of the `l3_main_clk` after reset, an additional wait state is necessary to access the boot ROM. Set the `waitstate` bit to add an additional wait state to the read access of the boot ROM. The enable safe mode warm reset update bit controls whether the wait state bit is updated during a warm reset.

Controlling the Boot Memory Map Through the System Interconnect

The System Manager contains a register, `noc_addr_remap_value`, that controls how the system interconnect maps memory addresses starting at 0x00000000. This register remaps the bottom of the memory map to either boot ROM or on-chip RAM.

Refer to "Address Remapping" in the System Interconnect chapter for details about using the `noc_addr_remap_value` register.

Related Information

Address Remapping on page 7-19

FPGA Interface Enables

The system manager can enable or disable interfaces between the FPGA and HPS.

The global interface bit (`intf`) of the global disable register (`gbl`) disables all interfaces between the FPGA and HPS.

Note: Ensure that the FPGA in configured before enabling the interfaces and that all interfaces between the FPGA and HPS are inactive before disabling them.

You can program the individual disable register (`indiv`) to disable the following interfaces between the FPGA and HPS:

You can program the FPGA interface enable registers (`fpgaintf_en_*`) to disable the following interfaces between the FPGA and HPS:

- Reset request interface
- JTAG enable interface
- I/O configuration interface
- Boundary scan interface
- Debug interface
- Trace interface
- System Trace Macrocell (STM) interface
- Cross-trigger interface (CTI)
- NAND interface
- SD/MMC interface
- SPI Master interface
- EMAC interfaces

ECC and Parity Control

The system manager can mask the ECC interrupts from each of the following HPS modules with ECC-protected RAM:

- MPU L2 cache data RAM
- On-chip RAM
- USB 2.0 OTG controller (USB0 and USB1) RAM
- EMAC (EMAC0, EMAC1, and EMAC2) RAM
- DMA controller RAM
- NAND flash controller RAM
- Quad SPI flash controller RAM
- SD/MMC controller RAM
- DDR interfaces

The system manager can inject single-bit or double-bit errors into the MPU L2 ECC memories for testing purposes. Set the bits in the appropriate memory enable register to inject errors. For example, to inject a single bit ECC error, set the `injs` bit of the `mpu_ctrl_l2_eccregister`.

Note: The injection request is edge-sensitive, meaning that the request is latched on 0 to 1 transitions on the injection bit. The next time a write operation occurs, the data will be corrupted, containing either a single or double bit error as selected. When the data is read back, the ECC logic detects the single or double bit error appropriately. The injection request cannot be cancelled, and the number of injections is limited to once every five MPU cycles.

The system manager can also inject parity failures into the parity-protected RAM in the MPU L2 to test the parity failure interrupt handler. Set the bits of the parity fail injection register (`parityinj`) to inject parity failures.

Note: Injecting parity failures into the parity-protected RAM in the MPU L2 causes the interrupt to be raised immediately. There is no actual error injected and the data is not corrupted. Furthermore, there is no need for a memory operation to actually be performed for the interrupt to be raised.

Preloader Handoff Information

The system manager provides eight 32-bit registers to store handoff information between the preloader and the operating system. The preloader can store any information in these registers. These register contents have no impact on the state of the HPS hardware. When the operating system kernel boots, it retrieves the information by reading the preloader to OS handoff information register array. These registers are reset only by a cold reset.

Clocks

The system manager is driven by a clock generated by the clock manager.

Related Information

Clock Manager on page 2-1

Resets

The system manager receives two reset signals from the reset manager. The `sys_manager_rst_n` signal is driven on a cold or warm reset and the `sys_manager_cold_rst_n` signal is driven only on a cold reset.

This function allows the system manager to reset some CSR fields on either a cold or warm reset and others only on a cold reset.

Related Information

Reset Manager on page 3-1

System Manager Address Map and Register Definitions

For complete HPS address map and register definitions, refer to the **Arria 10 HPS Address Map and Register Definitions**.

sys_mgr_core Address Map

Module Instance	Base Address	End Address
i_sys_mgr_core	0xFFD06000	0xFFD061FF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
siliconid1 on page 5-25	0x0	32	RO	0x10002	Silicon ID1 Register
siliconid2 on page 5-26	0x4	32	RO	0x0	Silicon ID2 Register
wddbg on page 5-27	0x8	32	RW	0xF	L4 Watchdog Debug Register
bootinfo on page 5-28	0xC	32	RO	0x0	Boot Info Register
mpu_ctrl_l2_ecc on page 5-31	0x10	32	RW	0x0	L2 Data RAM ECC Enable Register
dma on page 5-33	0x20	32	RW	0x100	Control Register
dma_periph on page 5-35	0x24	32	RW	0x0	Peripheral Security Register
sdmmc on page 5-35	0x28	32	RW	0x0	Control Register
sdmmc_l3master on page 5-37	0x2C	32	RW	0x3	SD/MMC L3 Master HPROT Register

Register	Offset	Width	Access	Reset Value	Description
nand_bootstrap on page 5-40	0x30	32	RW	0x0	Bootstrap Control Register
nand_l3master on page 5-41	0x34	32	RW	0x0	NAND L3 Master AxCACHE Register
usb0_l3master on page 5-43	0x38	32	RW	0x1	USB L3 Master HPROT Register
usb1_l3master on page 5-44	0x3C	32	RW	0x1	USB L3 Master HPROT Register
emac_global on page 5-46	0x40	32	RW	0x0	EMAC L3 Master AxCACHE Register
emac0 on page 5-47	0x44	32	RW	0x12000003	Control Register
emac1 on page 5-51	0x48	32	RW	0x12000003	Control Register
emac2 on page 5-55	0x4C	32	RW	0x12000003	Control Register
fpgaintf_en_global on page 5-60	0x60	32	RW	0x1	Global Disable Register
fpgaintf_en_0 on page 5-61	0x64	32	RW	0xFFFFFFFF	FPGA interface Individual Enable Register
fpgaintf_en_1 on page 5-63	0x68	32	RW	0xFFFFFFFF	FPGA interface Individual Enable Register
fpgaintf_en_2 on page 5-65	0x6C	32	RW	0x0	FPGA interface Individual Enable Register
fpgaintf_en_3 on page 5-67	0x70	32	RW	0x0	FPGA interface Individual Enable Register
noc_addr_remap_value on page 5-68	0x80	32	RW	0x0	Address remap register. This register drives the remap bits for the NOC. This is read / write register.

Register	Offset	Width	Access	Reset Value	Description
noc_addr_remap_set on page 5-69	0x84	32	WO	0x0	This is a Write 1 to Set register. Writing 0 is ignored, and writing 1 to a specific bit field sets the specific remap bit. Reads should not return an error, but the actual read value is "Undefined" .
noc_addr_remap_clear on page 5-70	0x88	32	WO	0x0	This is a Write 1 to Clear register. Writing 0 is ignored, and writing 1 to a specific bit field Clears the specific remap bit. Reads should not return an error, but the actual read value is "Undefined" .
ecc_intmask_value on page 5-71	0x90	32	RW	0x0	ECC interrupt mask register. This is a read/write register.
ecc_intmask_set on page 5-72	0x94	32	WO	0x0	ECC interrupt mask Set register
ecc_intmask_clr on page 5-73	0x98	32	WO	0x0	ECC interrupt mask Clear register
ecc_intstatus_serr on page 5-75	0x9C	32	RO	0x0	ECC single bit error status of individual modules. A write to this register should return an error.
ecc_intstatus_derr on page 5-76	0xA0	32	RO	0x0	ECC double bit error status of individual modules. A write to this register should return an error.

Register	Offset	Width	Access	Reset Value	Description
mpu_status_l2_ecc on page 5-77	0xA4	32	RO	0x0	This is a read only register which reads the current mpu L2 ecc interrupt status. A write to this register should return an error.
mpu_clear_l2_ecc on page 5-78	0xA8	32	RW	0x0	Write 1 to Clear register to clear the specific bit field of mpu l2 ecc interrupt pending status Reads should not return an error, but the read value is undefined.

Register	Offset	Width	Access	Reset Value	Description
mpu_status_l1_parity on page 5-79	0xAC	32	RO	0x0	Parity status from L1 and scu. This is a read only register. A write to this register should return an error. [17] CPU1 SCU parity error [16] CPU0 SCU parity error [15] CPU1 BTAC parity error [14] CPU1 GHB parity error [13] CPU1 instruction tag RAM parity error [12] CPU1 instruction data RAM parity error [11] CPU1 main TLB parity error [10] CPU1 data outer RAM parity error [9] CPU1 data tag RAM parity error [8] CPU1 data data RAM parity error. [7] CPU0 BTAC parity error [6] CPU0 GHB parity error [5] CPU0 instruction tag RAM parity error [4] CPU0 instruction data RAM parity error [3] CPU0 main TLB parity error [2] CPU0 data outer RAM parity error [1] CPU0 data tag RAM parity error [0] CPU0 data data RAM parity error.

Register	Offset	Width	Access	Reset Value	Description
mpu_clear_l1_parity on page 5-80	0xB0	32	RW	0x0	Parity status clear bit. A write to 1 of a specific bit clears the corresponding parity status bit. A read of this register should not return an error, but the actual read value is undefined. [17] CPU1 SCU parity error [16] CPU0 SCU parity error [15] CPU1 BTAC parity error [14] CPU1 GHB parity error [13] CPU1 instruction tag RAM parity error [12] CPU1 instruction data RAM parity error [11] CPU1 main TLB parity error [10] CPU1 data outer RAM parity error [9] CPU1 data tag RAM parity error [8] CPU1 data data RAM parity error. [7] CPU0 BTAC parity error [6] CPU0 GHB parity error [5] CPU0 instruction tag RAM parity error [4] CPU0 instruction data RAM parity error [3] CPU0 main TLB parity error [2] CPU0 data outer RAM parity error [1] CPU0 data tag RAM parity error [0] CPU0 data data RAM parity error.

Register	Offset	Width	Access	Reset Value	Description
mpu_set_l1_parity on page 5-82	0xB4	32	RW	0x0	Parity status set bit. A write to 1 of a specific bit sets the corresponding parity status bit. This register is used only to check the specific ISR routine. A read of this register should not return an error, but the actual read value is undefined. [17] CPU1 SCU parity error [16] CPU0 SCU parity error [15] CPU1 BTAC parity error [14] CPU1 GHB parity error [13] CPU1 instruction tag RAM parity error [12] CPU1 instruction data RAM parity error [11] CPU1 main TLB parity error [10] CPU1 data outer RAM parity error [9] CPU1 data tag RAM parity error [8] CPU1 data data RAM parity error. [7] CPU0 BTAC parity error [6] CPU0 GHB parity error [5] CPU0 instruction tag RAM parity error [4] CPU0 instruction data RAM parity error [3] CPU0 main TLB parity error [2] CPU0 data outer RAM parity error [1] CPU0 data tag RAM parity error [0] CPU0 data data RAM parity error.

Register	Offset	Width	Access	Reset Value	Description
noc_timeout on page 5-83	0xC0	32	RW	0x0	
noc_idlereq_set on page 5-84	0xC4	32	WO	0x0	Set IDLE request to each NOC master.
noc_idlereq_clr on page 5-85	0xC8	32	WO	0x0	Clear IDLE request to each NOC master.
noc_idlereq_value on page 5-86	0xCC	32	'not specified'	0x0	IDLE request to each NOC master. This register can be set by writing 1 to the specific bit in noc_idlereq_set register. This register can be cleared by writing 1 to the specific bit in noc_idlereq_clr register
noc_idleack on page 5-87	0xD0	32	RO	0x0	Idle acknowledge value from NOC Masters. This is asserted (value 1 in the field) in response to the IDLE requests asserted by software.
noc_idlestatus on page 5-88	0xD4	32	RO	0x0	Status of IDLE from the NOC masters. A 1 in the field means the specific master is idle.
fpga2soc_ctrl on page 5-89	0xD8	32	RW	0x0	

sys_mgr_core Summary

Base Address: 0xFFD06000

Register Address Offset	Bit Fields															
i_sys_mgr_core																
siliconid1 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	id RO 0x1															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rev RO 0x2															
siliconid2 0x4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	rsv RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsv RO 0x0															
wddbg 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved												mode_1 RW 0x3		mode_0 RW 0x3	
bootinfo 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	bsel RO 0x0			Reserved	pin_bsel RO 0x0			Reserved	fpga_bsel RO 0x0			Reserved			fpga_bsel_en RO 0x0
mpu_ctrl_l2_ecc 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															inj_type RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							inj_en RW 0x0	Reserved							ecc_en RW 0x0

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	irq_ns RW 0x0								Reserved							mgr_ns RW 0x0
dma 0x20	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							chansel_2 RW 0x1	Reserved			chansel_1 RW 0x0	Reserved			chansel_0 RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ns RW 0x0															
dma_periph 0x24	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ns RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
sdmmc 0x28	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved									smplsel RW 0x0			Reserved	drvsel RW 0x0		
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
sdmmc_l3master 0x2C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved											hprot RW 0x3				
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved			page512_x16 RW 0x0	Reserved			page512 RW 0x0	Reserved							tworowaddr RW 0x0
nand_bootstrap 0x30	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							noloadb0p0 RW 0x0	Reserved							noinit RW 0x0

Register Address Offset	Bit Fields															
nand_l3master 0x34	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								awcache_0 RW 0x0				arcache_0 RW 0x0			
usb0_l3master 0x38	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved												hprot RW 0x1			
usb1_l3master 0x3C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved												hprot RW 0x1			
emac_global 0x40	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															ptp_clk_sel RW 0x0
emac0 0x44	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	axi_disable RW 0x0	sbd_data_endianness RW 0x0	Reserved	awprot RW 0x2		Reserved	arprot RW 0x2		awcache RW 0x0				arcache RW 0x0			
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							ptp_ref_sel RW 0x0	Reserved						phy_intf_sel RW 0x3	

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
emac1 0x48	axi_disable RW 0x0	sbd_data_endianness RW 0x0	Reserved	awprot RW 0x2		Reserved	arprot RW 0x2		awcache RW 0x0				arcache RW 0x0			
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							ptp_ref_sel RW 0x0	Reserved						phy_intf_sel RW 0x3	
emac2 0x4C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	axi_disable RW 0x0	sbd_data_endianness RW 0x0	Reserved	awprot RW 0x2		Reserved	arprot RW 0x2		awcache RW 0x0				arcache RW 0x0			
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							ptp_ref_sel RW 0x0	Reserved						phy_intf_sel RW 0x3	
fpgaintf_en_global 0x60	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															intf RW 0x1
fpgaintf_en_0 0x64	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															bscan RW 0x1
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							cfgio RW 0x1	Reserved							rstreq RW 0x1

Register Address Offset	Bit Fields															
fpgaintf_en_1 0x68	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved							ctmtrigger RW 0x1	Reserved							stmevent RW 0x1
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							dbgapb RW 0x1	Reserved			trace RW 0x1	Reserved			
fpgaintf_en_2 0x6C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved							spim_1 RW 0x0	Reserved							spim_0 RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							sdmmc RW 0x0	Reserved			nand RW 0x0	Reserved			
fpgaintf_en_3 0x70	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															emac_2 RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							emac_1 RW 0x0	Reserved							emac_0 RW 0x0
noc_addr_remap_value 0x80	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														remap1 0x0	remap0 0x0
noc_addr_remap_set 0x84	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														remap1 0x0	remap0 0x0

Register Address Offset	Bit Fields															
noc_addr_remap_clear 0x88	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														remap1 0x0	remap0 0x0
ecc_intmask_value 0x90	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved													ddr1 0x0	ddr0 0x0	sdmmcb 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	sdmmca 0x0	qspi 0x0	nand_rd 0x0	nand_wr 0x0	nand_buf 0x0	dma 0x0	emac2_tx 0x0	emac2_rx 0x0	emac1_tx 0x0	emac1_rx 0x0	emac0_tx 0x0	emac0_rx 0x0	usb1 0x0	usb0 0x0	ocram 0x0	l2 0x0
ecc_intmask_set 0x94	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved													ddr1 0x0	ddr0 0x0	sdmmcb 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	sdmmca 0x0	qspi 0x0	nand_rd 0x0	nand_wr 0x0	nand_buf 0x0	dma 0x0	emac2_tx 0x0	emac2_rx 0x0	emac1_tx 0x0	emac1_rx 0x0	emac0_tx 0x0	emac0_rx 0x0	usb1 0x0	usb0 0x0	ocram 0x0	l2 0x0
ecc_intmask_clr 0x98	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved													ddr1 0x0	ddr0 0x0	sdmmcb 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	sdmmca 0x0	qspi 0x0	nand_rd 0x0	nand_wr 0x0	nand_buf 0x0	dma 0x0	emac2_tx 0x0	emac2_rx 0x0	emac1_tx 0x0	emac1_rx 0x0	emac0_tx 0x0	emac0_rx 0x0	usb1 0x0	usb0 0x0	ocram 0x0	l2 0x0
ecc_intstatus_serr 0x9C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved													ddr1 0x0	ddr0 0x0	sdmmcb 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	sdmmca 0x0	qspi 0x0	nand_rd 0x0	nand_wr 0x0	nand_buf 0x0	dma 0x0	emac2_tx 0x0	emac2_rx 0x0	emac1_tx 0x0	emac1_rx 0x0	emac0_tx 0x0	emac0_rx 0x0	usb1 0x0	usb0 0x0	ocram 0x0	l2 0x0

Register Address Offset	Bit Fields															
ecc_intstatus_derr 0xA0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved													ddr1 0x0	ddr0 0x0	sdmmcb 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	sdmmca 0x0	qspi 0x0	nand_rd 0x0	nand_wr 0x0	nand_buf 0x0	dma 0x0	emac2_tx 0x0	emac2_rx 0x0	emac1_tx 0x0	emac1_rx 0x0	emac0_tx 0x0	emac0_rx 0x0	usb1 0x0	usb0 0x0	ocram 0x0	l2 0x0
mpu_status_l2_ecc 0xA4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	merr_pending 0x0	Reserved			merr_info 0x0											
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	serr_pending 0x0	Reserved			serr_info 0x0											
mpu_clear_l2_ecc 0xA8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	merr 0x0	Reserved														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	serr 0x0	Reserved														
mpu_status_l1_parity 0xAC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved														scu 0x0	
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cpu1 0x0								cpu0 0x0							
mpu_clear_l1_parity 0xB0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved														scu 0x0	
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cpu1 0x0								cpu0 0x0							

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
mpu_set_l1_parity 0xB4	Reserved														scu 0x0	
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cpu1 0x0								cpu0 0x0							
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
noc_timeout 0xC0	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															en 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
noc_idlereq_set 0xC4	Reserved							fpga2sdram2 0x0	Reserved			fpga2sdram1 0x0	Reserved			fpga2sdram0 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							fpga2soc 0x0	Reserved			lwsoc2fpga 0x0	Reserved			soc2fpga 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
noc_idlereq_clr 0xC8	Reserved							fpga2sdram2 0x0	Reserved			fpga2sdram1 0x0	Reserved			fpga2sdram0 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							fpga2soc 0x0	Reserved			lwsoc2fpga 0x0	Reserved			soc2fpga 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
noc_idlereq_value 0xCC	Reserved							fpga2sdram2 0x0	Reserved			fpga2sdram1 0x0	Reserved			fpga2sdram0 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							fpga2soc 0x0	Reserved			lwsoc2fpga 0x0	Reserved			soc2fpga 0x0

Register Address Offset	Bit Fields															
noc_idleack 0xD0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved							fpga2sdram2 0x0	Reserved			fpga2sdram1 0x0	Reserved			fpga2sdram0 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							fpga2soc 0x0	Reserved			lwsoc2fpga 0x0	Reserved			soc2fpga 0x0
noc_idlestatus 0xD4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved							fpga2sdram2 0x0	Reserved			fpga2sdram1 0x0	Reserved			fpga2sdram0 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							fpga2soc 0x0	Reserved			lwsoc2fpga 0x0	Reserved			soc2fpga 0x0
fpga2soc_ctrl 0xD8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															allow_secure 0x0

siliconid1

Specifies Silicon ID and revision number.
This is a read only register and a write should return error.

Module Instance	Base Address	Register Address
i_sys_mgr_core	0xFFD06000	0xFFD06000

Offset: 0x0

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
id RO 0x1															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rev RO 0x1															

siliconid1 Fields

Bit	Name	Description	Access	Reset
31:16	id	Silicon ID **Description** HPS_NF	RO	0x1
15:0	rev	Silicon revision number. **Description** rev2	RO	0x2

siliconid2

```
Reserved for future use.
This is a read only register and a write should return error.
```

Module Instance	Base Address	Register Address
i_sys_mgr_core	0xFFD06000	0xFFD06004

Offset: 0x4

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
rsv RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rsv RO 0x0															

siliconid2 Fields

Bit	Name	Description	Access	Reset
31:0	rsv	Reserved for future use.	RO	0x0

wddbg

Controls the behavior of the L4 watchdogs when the CPUs are in debug mode. These control registers are used to drive the pause input signal of the L4 watchdogs. Note that the watchdogs built into the MPU automatically are paused when their associated CPU enters debug mode. Only reset by a cold reset.

Module Instance	Base Address	Register Address
i_sys_mgr_core	0xFFD06000	0xFFD06008

Offset: 0x8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												mode_1 RW 0x3		mode_0 RW 0x3	

wddbg Fields

Bit	Name	Description	Access	Reset
3:2	mode_1	Controls behavior of L4 watchdog when CPUs in debug mode. Field array index matches L4 watchdog index. Value: Description 0: Continue 1: PauseCPU0 2: PauseCPU1 3: PauseEither	RW	0x3
1:0	mode_0	Controls behavior of L4 watchdog when CPUs in debug mode. Field array index matches L4 watchdog index. Value: Description 0: Continue 1: PauseCPU0 2: PauseCPU1 3: PauseEither	RW	0x3

bootinfo

```
        Provides access to boot configuration information.
This is a read only register and a write should return error.
This register gets reset only on a cold reset.
```

Module Instance	Base Address	Register Address
i_sys_mgr_core	0xFFD06000	0xFFD0600C

Offset: 0xC

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	bsel RO 0x0			Reserved	pin_bsel RO 0x0			Reserved	fpga_bsel RO 0x0			Reserved			fpga_bsel_en RO 0x0

bootinfo Fields

Bit	Name	Description	Access	Reset
14:12	bsel	Multiplexed Value of Boot Select from pins and fpga **Value** **Description** 0 RESERVEDx 1 FPGA 2 NAND_Flash_1_8v 3 NAND_Flash_3_0v 4 SD_MMC_EXTERNAL_TRANSCEIVER_1_8v 5 SD_MMC_INTERNAL_TRANSCEIVER_3_0v 6 QSPI_Flash_1_8v 7 QSPI_Flash_3_0v	RO	0x0

Bit	Name	Description	Access	Reset
10:8	pin_bsel	Specifies the sampled value of the HPS BSEL pins. The value of HPS BSEL pins are sampled upon deassertion of cold reset. Value — Description 0 — RESERVEDx 1 — FPGA 2 — NAND_Flash_1_8v 3 — NAND_Flash_3_0v 4 — SD_MMC_EXTERNAL_TRANSCEIVER_1_8v 5 — SD_MMC_INTERNAL_TRANSCEIVER_3_0v 6 — QSPI_Flash_1_8v 7 — QSPI_Flash_3_0v	RO	0x0

Bit	Name	Description	Access	Reset
6:4	fpga_bsel	The boot select field specifies the boot source. It is read by the Boot ROM code on a cold or warm reset to determine the boot source. The boot select value is equal to the f2s_bsel signal from the FPGA if the f2s_bsel_en signal from the FPGA is 1 or equal to the sampled value of HPS BSEL pins if the f2s_bsel_en signal from the FPGA is 0 or the FPGA is not powered on or not in User Mode (fpga_config_complete = 0). The HPS BSEL pins value are sampled upon deassertion of cold reset. **Value** **Description** 0 RESERVEDx 1 FPGA 2 NAND_Flash_1_8v 3 NAND_Flash_3_0v 4 SD_MMC_EXTERNAL_TRANSCEIVER_1_8v 5 SD_MMC_INTERNAL_TRANSCEIVER_3_0v 6 QSPI_Flash_1_8v 7 QSPI_Flash_3_0v	RO	0x0
0	fpga_bsel_en	Specifies the value of the f2s_bsel_en. f2s_bsel_en is a signal from FPGA. If 1, boot select value is equal to FPGA boot select signal (f2s_bsel). If 0, boot select value is equal to the sampled value of HPS BSEL pins. Value of f2s_bsel_en is overidden to 0x0 if FPGA is not in user mode (fpga_config_complete = 0)	RO	0x0

mpu_ctrl_l2_ecc

This register is used to enable ECC on the L2 Data RAM. ECC errors can be injected into the write path using bits in this register. This register contains interrupt status of the ECC single/

double bit error.
Some fileds of this register are only reset by a cold reset (ignores warm reset).
Some fields are affected by both warm and cold reset.

Module Instance	Base Address	Register Address
i_sys_mgr_core	0xFFD06000	0xFFD06010

Offset: 0x10

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															inj_type RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							inj_en RW 0x0	Reserved							ecc_en RW 0x0

mpu_ctrl_l2_ecc Fields

Bit	Name	Description	Access	Reset
16	inj_type	MPU L2 ECC error injection type. This bit will get reset on a warm reset and cold reset. Value: Description 0: single_bit 1: double_bit	RW	0x0

Bit	Name	Description	Access	Reset
8	inj_en	Error injection enable. Write 1 here to enable error injection to MPU L2. Please note that if ECC is not enabled by writing 1 to ecc_en bit there wont be any error injections. This bit will get reset on a warm reset and cold reset.	RW	0x0
0	ecc_en	Enable Single bit or Double bit error Detection and Single bit Error Correction for L2 Data RAM Only reset by a cold reset (ignores warm reset).	RW	0x0

dma

Registers used by the DMA Controller. All fields are reset by a cold or warm reset.
These register bits should be updated during system initialization prior to removing the DMA controller from reset. They may not be changed dynamically during DMA operation.

Module Instance	Base Address	Register Address
i_sys_mgr_core	0xFFD06000	0xFFD06020

Offset: 0x20

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
irq_ns RW 0x0								Reserved							mgr_ns RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							chansel_2 RW 0x1	Reserved			chansel_1 RW 0x0	Reserved			chansel_0 RW 0x0

dma Fields

Bit	Name	Description	Access	Reset
31:24	irq_ns	Specifies the security state of an event-interrupt resource. If bit index [x] is 0, the DMAC assigns event<x> or irq[x] to the Secure state. If bit index [x] is 1, the DMAC assigns event<x> or irq[x] to the Non-secure state.	RW	0x0
16	mgr_ns	Specifies the security state of the DMA manager thread. 0 = assigns DMA manager to the Secure state. 1 = assigns DMA manager to the Non-secure state. Sampled by the DMA controller when it exits from reset.	RW	0x0
8	chansel_2	select between FPGA interface 5 and Security Manager to be mapped to DMA peripheral request index 5 **Value** **Description** 0 FPGA 1 SECMGR	RW	0x1
4	chansel_1	select between FPGA interface 7 and I2C4_Rx to be mapped to DMA peripheral request index 7 **Value** **Description** 0 FPGA 1 I2C4_RX	RW	0x0
0	chansel_0	Select between FPGA interface 6 and I2C4_Tx to be mapped to DMA peripheral request index 6 **Value** **Description** 0 FPGA 1 I2C4_TX	RW	0x0

dma_periph

Controls the security state of a peripheral request interface. Sampled by the DMA controller when it exits from reset. These register bits should be updated during system initialization prior to removing the DMA controller from reset. They may not be changed dynamically during DMA operation.

Module Instance	Base Address	Register Address
i_sys_mgr_core	0xFFD06000	0xFFD06024

Offset: 0x24

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ns RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ns RW 0x0															

dma_periph Fields

Bit	Name	Description	Access	Reset
31:0	ns	If bit index [x] is 0, the DMA controller assigns peripheral request interface x to the Secure state. If bit index [x] is 1, the DMA controller assigns peripheral request interface x to the Non-secure state. Reset by a cold or warm reset.	RW	0x0

sdmmc

Registers used by the SDMMC Controller. All fields are reset by a cold or warm reset.

Module Instance	Base Address	Register Address
i_sys_mgr_core	0xFFD06000	0xFFD06028

Offset: 0x28

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									smplsel RW 0x0			Reserved	drvsel RW 0x0		

sdmmc Fields

Bit	Name	Description	Access	Reset
6:4	smplsel	Select which phase shift of the clock for cclk_in_sample. Note that the boot ROM programs this field to 0x0. Value — Description 0 — degrees0 1 — degrees45 2 — degrees90 3 — degrees135 4 — degrees180 5 — degrees225 6 — degrees270 7 — degrees315	RW	0x0

Bit	Name	Description	Access	Reset
2:0	drvsel	Select which phase shift of the clock for cclk_in_drv. Note that the boot ROM programs this field to 0x3. Value: Description 0: Reserved 1: degrees45 2: degrees90 3: degrees135 4: degrees180 5: degrees225 6: degrees270 7: degrees315	RW	0x0

sdmmc_l3master

Controls the L3 master HPROT AHB-Lite signal.
These register bits should be updated only during system initialization prior to removing the peripheral from reset. They may not be changed dynamically during peripheral operation
All fields are reset by a cold or warm reset.

Module Instance	Base Address	Register Address
i_sys_mgr_core	0xFFD06000	0xFFD0602C

Offset: 0x2C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved											hprot RW 0x3				

sdmmc_l3master Fields

Bit	Name	Description	Access	Reset
4:0	hprot	========== HPROT[4] Allocate 0: L3 master accesses for the module are non-allocatable 1: L3 master accesses for the module are allocatable ========== HPROT[3] Cachable 0: L3 master accesses for the module are non-cacheable. 1: L3 master accesses for the module are cacheable. ========== HPROT[2] Bufferable 0: L3 master accesses for the module are not bufferable. 1: L3 master accesses for the module are bufferable. ========== HPROT[1] Privileged 0: L3 master accesses for the module are not privileged. 1: L3 master accesses for the module are privileged. ========== HPROT[0] Data/Opcode 0: Specifies if the L3 master access is for opcode 1: Specifies if the L3 master access is for data ========== HPROT[4:2] Example Encodings ========== 0 0 0 Strongly Ordered, cannot be buffered 0 0 1 Device, can be buffered 0 1 0 Cachable (Outer Noncachable, do not allocate on reads or writes) 1 1 0 Cachable Write-Through (allocate on reads only, no allocate on write) 0 1 1 Cachable Write-Back (allocate on reads and writes) 1 1 1 Cachable Write-Back (allocate on reads only, no allocate on write) ========== HPROT[1:0] Example Encodings ========== 0 - Opcode Fetch 0 - User Access 1 - Privileged Access ==========	RW	0x3

nand_bootstrap

Bootstrap fields sampled by NAND Flash Controller when released from reset.
All fields are reset by a cold or warm reset.

Module Instance	Base Address	Register Address
i_sys_mgr_core	0xFFD06000	0xFFD06030

Offset: 0x30

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved			page512_x16 RW 0x0	Reserved			page512 RW 0x0	Reserved							tworowaddr RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							noloadb0p0 RW 0x0	Reserved							noinit RW 0x0

nand_bootstrap Fields

Bit	Name	Description	Access	Reset
28	page512_x16	Reset value - 0 Field name: PAGE512_x16_DEVICE Description: If 1, NAND device has 512 bytes page size and I/O width is 16 bits. This start should be asserted in case of 512 bytes devices only. This signal must be stable and have proper value by the time Controller comes out of Reset	RW	0x0
24	page512	If 1, NAND device has a 512 byte page size.	RW	0x0

Bit	Name	Description	Access	Reset
16	tworowaddr	If 1, NAND device requires only 2 row address cycles instead of the normal 3 row address cycles.	RW	0x0
8	noloadb0p0	If 1, inhibits NAND Flash Controller from loading page 0 of block 0 of the NAND device as part of the initialization procedure.	RW	0x0
0	noinit	If 1, inhibits NAND Flash Controller from performing initialization when coming out of reset. Instead, software must program all registers pertaining to device parameters like page size, width, etc.	RW	0x0

nand_l3master

Controls the L3 master ARCACHE and AWCACHE AXI signals. These register bits should be updated only during system initialization prior to removing the peripheral from reset. They may not be changed dynamically during peripheral operation
All fields are reset by a cold or warm reset.

Module Instance	Base Address	Register Address
i_sys_mgr_core	0xFFD06000	0xFFD06034

Offset: 0x34

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								awcache_0 RW 0x0				arcache_0 RW 0x0			

nand_l3master Fields

Bit	Name	Description	Access	Reset
7:4	awcache_0	Specifies the value of the module AWCACHE signal. Value — Description 0 — Noncache_Nonbuff 1 — Buff 2 — Cache_Nonalloc 3 — Cache_Buff_Nonalloc 4 — Reserved1 5 — Reserved2 6 — Cache_Wrthru_Rdalloc 7 — Cache_Wrback_Rdalloc 8 — Reserved3 9 — Reserved4 10 — Cache_Wrthru_Wralloc 11 — Cache_Wrback_Wralloc 12 — Reserved5 13 — Reserved6 14 — Cache_Wrthru_Alloc 15 — Cache_Wrback_Alloc	RW	0x0

Bit	Name	Description	Access	Reset
3:0	arcache_0	Specifies the value of the module ARCACHE signal. Value — Description 0 — Noncache_Nonbuff 1 — Buff 2 — Cache_Nonalloc 3 — Cache_Buff_Nonalloc 4 — Reserved1 5 — Reserved2 6 — Cache_Wrthru_Rdalloc 7 — Cache_Wrback_Rdalloc 8 — Reserved3 9 — Reserved4 10 — Cache_Wrthru_Wralloc 11 — Cache_Wrback_Wralloc 12 — Reserved5 13 — Reserved6 14 — Cache_Wrthru_Alloc 15 — Cache_Wrback_Alloc	RW	0x0

usb0_l3master

Controls the L3 master HPROT AHB-Lite signal.
These register bits should be updated only during system initialization prior to removing the peripheral from reset. They may not be changed dynamically during peripheral operation
All fields are reset by a cold or warm reset.

Module Instance	Base Address	Register Address
i_sys_mgr_core	0xFFD06000	0xFFD06038

Offset: 0x38

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												hprot RW 0x1			

usb0_l3master Fields

Bit	Name	Description	Access	Reset
3:0	hprot	Defines HPROT[4:1]. HPROT[0] from usb is tied HIGH allow only data access. ========== HPROT[4] Allocate 0: L3 master accesses for the module are non-allocatable 1: L3 master accesses for the module are allocatable ========== HPROT[3] Cachable 0: L3 master accesses for the module are non-cacheable. 1: L3 master accesses for the module are cacheable. ========== HPROT[2] Bufferable 0: L3 master accesses for the module are not bufferable. 1: L3 master accesses for the module are bufferable. ========== HPROT[1] Privileged 0: L3 master accesses for the module are not privileged. 1: L3 master accesses for the module are privileged. ==========	RW	0x1

usb1_l3master

Controls the L3 master HPROT AHB-Lite signal. These register bits should be updated only during system initialization prior to removing the peripheral from reset. They may not be

changed dynamically during peripheral operation
All fields are reset by a cold or warm reset.

Module Instance	Base Address	Register Address
i_sys_mgr_core	0xFFD06000	0xFFD0603C

Offset: 0x3C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												hprot RW 0x1			

usb1_l3master Fields

Bit	Name	Description	Access	Reset
3:0	hprot	Defines HPROT[4:1]. HPROT[0] from usb is tied HIGH to allow only data access. ========== HPROT[4] Allocate 0: L3 master accesses for the module are non-allocatable 1: L3 master accesses for the module are allocatable ========== HPROT[3] Cachable 0: L3 master accesses for the module are non-cacheable. 1: L3 master accesses for the module are cacheable. ========== HPROT[2] Bufferable 0: L3 master accesses for the module are not bufferable. 1: L3 master accesses for the module are bufferable. ========== HPROT[1] Privileged 0: L3 master accesses for the module are not privileged. 1: L3 master accesses for the module are privileged. ==========	RW	0x1

emac_global

Controls the L3 master ARCACHE and AWCACHE AXI signals. These register bits should be updated only during system initialization prior to removing the peripheral from reset. They may not be changed dynamically during peripheral operation
All fields are reset by a cold or warm reset.

Module Instance	Base Address	Register Address
i_sys_mgr_core	0xFFD06000	0xFFD06040

Offset: 0x40

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															ptp_clk_sel RW 0x0

emac_global Fields

Bit	Name	Description	Access	Reset
0	ptp_clk_sel	Selects the source of the PTP reference clock between emac_ptp_clk from the Clock Manager or f2s_ptp_ref_clk from the FPGA Fabric. Value: 0 — emac_ptp_clk; 1 — f2s_ptp_ref_clk	RW	0x0

emac0

Registers used by the EMAC. All fields are reset by a cold or warm reset.

Module Instance	Base Address	Register Address
i_sys_mgr_core	0xFFD06000	0xFFD06044

Offset: 0x44

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
axi_disable RW 0x0	Reserved		awprot RW 0x2		Reserved	arprot RW 0x2		awcache RW 0x0				arcache RW 0x0			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							ptp_ref_sel RW 0x0	Reserved						phy_intf_sel RW 0x3	

emac0 Fields

Bit	Name	Description	Access	Reset
31	axi_disable	AXI Disable	RW	0x0
28:27	awprot	Specifies the values of the 2 EMAC AWCACHE signals. ========== AxPROT[1] LOW: Secure Access HIGH: NonSecure Access ========== AxPROT[0] LOW: Normal Access HIGH: Privileged Access ========== **Value** **Description** 0 Secure Normal(non-privileged) access 1 Secure Privileged access 2 Non-Secure Normal(non-privileged) access 3 Non-Secure Privileged access	RW	0x2

Bit	Name	Description	Access	Reset
25:24	arprot	Specifies the values of the ARPROT signals. ========== AxPROT[1] LOW: Secure Access HIGH: NonSecure Access ========== AxPROT[0] LOW: Normal Access HIGH: Privileged Access ========== **Value** **Description** 0 Secure Normal(non-privileged) access 1 Secure Privileged access 2 Non-Secure Normal(non-privileged) access 3 Non-Secure Privileged access	RW	0x2
23:20	awcache	Specifies the values of the 2 EMAC AWCACHE signals. The field array index corresponds to the EMAC index. **Value** **Description** 0 Noncache_Nonbuff 1 Buff 2 Cache_Nonalloc 3 Cache_Buff_Nonalloc 4 Reserved1 5 Reserved2 6 Cache_Wrthru_Rdalloc 7 Cache_Wrback_Rdalloc 8 Reserved3 9 Reserved4 10 Cache_Wrthru_Wralloc 11 Cache_Wrback_Wralloc 12 Reserved5 13 Reserved6 14 Cache_Wrthru_Alloc 15 Cache_Wrback_Alloc	RW	0x0

Bit	Name	Description	Access	Reset
19:16	arcache	Specifies the values of the 2 EMAC ARCACHE signals. The field array index corresponds to the EMAC index. Value — Description 0 — Noncache_Nonbuff 1 — Buff 2 — Cache_Nonalloc 3 — Cache_Buff_Nonalloc 4 — Reserved1 5 — Reserved2 6 — Cache_Wrthru_Rdalloc 7 — Cache_Wrback_Rdalloc 8 — Reserved3 9 — Reserved4 10 — Cache_Wrthru_Wralloc 11 — Cache_Wrback_Wralloc 12 — Reserved5 13 — Reserved6 14 — Cache_Wrthru_Alloc 15 — Cache_Wrback_Alloc	RW	0x0
8	ptp_ref_sel	This field selects if the Timestamp reference is internally or externally generated. EMAC0 may be the master to generate the timestamp for EMAC1 and EMAC2. EMAC0 must be set to Internal Timestamp. EMAC1/2 may be set to either Internal or External. Value — Description 0 — internal 1 — external	RW	0x0

Bit	Name	Description	Access	Reset
1:0	phy_intf_sel	PHY Interface Select Field to select "Out of Reset", GMII (or MII), RGMII or RMII as the PHY interface. Note, the MAC speed is an output of Synopsys IP. So the System Manager PHY Select combined with MAC speed from the IP determine the clock/PHY configuration. "Out of Reset" mode implies that the MAC RX and TX internal clocks use the Clock Manager reference rather than depending on the PHY to have active clocks. **Value** **Description** 0 GMII_MII 1 RGMII 2 RMII 3 RESET	RW	0x3

emac1

Registers used by the EMAC. All fields are reset by a cold or warm reset.

Module Instance	Base Address	Register Address
i_sys_mgr_core	0xFFD06000	0xFFD06048

Offset: 0x48

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
axi_disable RW 0x0	Reserved		awprot RW 0x2		Reserved	arprot RW 0x2		awcache RW 0x0				arcache RW 0x0			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							ptp_ref_sel RW 0x0	Reserved						phy_intf_sel RW 0x3	

emac1 Fields

Bit	Name	Description	Access	Reset
31	axi_disable	AXI Disable	RW	0x0
28:27	awprot	Specifies the values of the 2 EMAC AWCACHE signals. ========== AxPROT[1] LOW: Secure Access HIGH: NonSecure Access ========== AxPROT[0] LOW: Normal Access HIGH: Privileged Access ========== **Value** **Description** 0 Secure Normal(non-privileged) access 1 Secure Privileged access 2 Non-Secure Normal(non-privileged) access 3 Non-Secure Privileged access	RW	0x2

Bit	Name	Description	Access	Reset
25:24	arprot	Specifies the values of the ARPROT signals. ========== AxPROT[1] LOW: Secure Access HIGH: NonSecure Access ========== AxPROT[0] LOW: Normal Access HIGH: Privileged Access ========== **Value** **Description** 0 Secure Normal(non-privileged) access 1 Secure Privileged access 2 Non-Secure Normal(non-privileged) access 3 Non-Secure Privileged access	RW	0x2
23:20	awcache	Specifies the values of the 2 EMAC AWCACHE signals. The field array index corresponds to the EMAC index. **Value** **Description** 0 Noncache_Nonbuff 1 Buff 2 Cache_Nonalloc 3 Cache_Buff_Nonalloc 4 Reserved1 5 Reserved2 6 Cache_Wrthru_Rdalloc 7 Cache_Wrback_Rdalloc 8 Reserved3 9 Reserved4 10 Cache_Wrthru_Wralloc 11 Cache_Wrback_Wralloc 12 Reserved5 13 Reserved6 14 Cache_Wrthru_Alloc 15 Cache_Wrback_Alloc	RW	0x0

Bit	Name	Description	Access	Reset
19:16	arcache	Specifies the values of the 2 EMAC ARCACHE signals. The field array index corresponds to the EMAC index. Value — Description 0 — Noncache_Nonbuff 1 — Buff 2 — Cache_Nonalloc 3 — Cache_Buff_Nonalloc 4 — Reserved1 5 — Reserved2 6 — Cache_Wrthru_Rdalloc 7 — Cache_Wrback_Rdalloc 8 — Reserved3 9 — Reserved4 10 — Cache_Wrthru_Wralloc 11 — Cache_Wrback_Wralloc 12 — Reserved5 13 — Reserved6 14 — Cache_Wrthru_Alloc 15 — Cache_Wrback_Alloc	RW	0x0
8	ptp_ref_sel	This field selects if the Timestamp reference is internally or externally generated. EMAC0 may be the master to generate the timestamp for EMAC1 and EMAC2. EMAC0 must be set to Internal Timestamp. EMAC1/2 may be set to either Internal or External. Value — Description 0 — internal 1 — external	RW	0x0

Bit	Name	Description	Access	Reset
1:0	phy_intf_sel	PHY Interface Select Field to select "Out of Reset", GMII (or MII), RGMII or RMII as the PHY interface. Note, the MAC speed is an output of Synopsys IP. So the System Manager PHY Select combined with MAC speed from the IP determine the clock/PHY configuration. "Out of Reset" mode implies that the MAC RX and TX internal clocks use the Clock Manager reference rather than depending on the PHY to have active clocks. **Value** **Description** 0 GMII_MII 1 RGMII 2 RMII 3 RESET	RW	0x3

emac2

Registers used by the EMAC. All fields are reset by a cold or warm reset.

Module Instance	Base Address	Register Address
i_sys_mgr_core	0xFFD06000	0xFFD0604C

Offset: 0x4C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
axi_disable RW 0x0	Reserved		awprot RW 0x2		Reserved	arprot RW 0x2		awcache RW 0x0				arcache RW 0x0			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							ptp_ref_sel RW 0x0	Reserved						phy_intf_sel RW 0x3	

emac2 Fields

Bit	Name	Description	Access	Reset
31	axi_disable	AXI Disable	RW	0x0
28:27	awprot	Specifies the values of the 2 EMAC AWCACHE signals. ========== AxPROT[1] LOW: Secure Access HIGH: NonSecure Access ========== AxPROT[0] LOW: Normal Access HIGH: Privileged Access ========== **Value** — **Description** 0 — Secure Normal(non-privileged) access 1 — Secure Privileged access 2 — Non-Secure Normal(non-privileged) access 3 — Non-Secure Privileged access	RW	0x2

Bit	Name	Description	Access	Reset
25:24	arprot	Specifies the values of the ARPROT signals. ========== AxPROT[1] LOW: Secure Access HIGH: NonSecure Access ========== AxPROT[0] LOW: Normal Access HIGH: Privileged Access ========== **Value** / **Description** 0 — Secure Normal(non-privileged) access 1 — Secure Privileged access 2 — Non-Secure Normal(non-privileged) access 3 — Non-Secure Privileged access	RW	0x2

Bit	Name	Description	Access	Reset
23:20	awcache	Specifies the values of the 2 EMAC AWCACHE signals. The field array index corresponds to the EMAC index. **Value** — **Description** 0 — Noncache_Nonbuff 1 — Buff 2 — Cache_Nonalloc 3 — Cache_Buff_Nonalloc 4 — Reserved1 5 — Reserved2 6 — Cache_Wrthru_Rdalloc 7 — Cache_Wrback_Rdalloc 8 — Reserved3 9 — Reserved4 10 — Cache_Wrthru_Wralloc 11 — Cache_Wrback_Wralloc 12 — Reserved5 13 — Reserved6 14 — Cache_Wrthru_Alloc 15 — Cache_Wrback_Alloc	RW	0x0

Bit	Name	Description	Access	Reset
19:16	arcache	Specifies the values of the 2 EMAC ARCACHE signals. The field array index corresponds to the EMAC index. **Value** — **Description** 0 — Noncache_Nonbuff 1 — Buff 2 — Cache_Nonalloc 3 — Cache_Buff_Nonalloc 4 — Reserved1 5 — Reserved2 6 — Cache_Wrthru_Rdalloc 7 — Cache_Wrback_Rdalloc 8 — Reserved3 9 — Reserved4 10 — Cache_Wrthru_Wralloc 11 — Cache_Wrback_Wralloc 12 — Reserved5 13 — Reserved6 14 — Cache_Wrthru_Alloc 15 — Cache_Wrback_Alloc	RW	0x0
8	ptp_ref_sel	This field selects if the Timestamp reference is internally or externally generated. EMAC0 may be the master to generate the timestamp for EMAC1 and EMAC2. EMAC0 must be set to Internal Timestamp. EMAC1/2 may be set to either Internal or External. **Value** — **Description** 0 — internal 1 — external	RW	0x0

Bit	Name	Description	Access	Reset
1:0	phy_intf_sel	PHY Interface Select Field to select "Out of Reset", GMII (or MII), RGMII or RMII as the PHY interface. Note, the MAC speed is an output of Synopsys IP. So the System Manager PHY Select combined with MAC speed from the IP determine the clock/PHY configuration. "Out of Reset" mode implies that the MAC RX and TX internal clocks use the Clock Manager reference rather than depending on the PHY to have active clocks. **Value** — **Description** 0 — GMII_MII 1 — RGMII 2 — RMII 3 — RESET	RW	0x3

fpgaintf_en_global

Used to disable all interfaces between the FPGA and HPS. This register is reset only on a cold reset (ignores warm reset).

Module Instance	Base Address	Register Address
i_sys_mgr_core	0xFFD06000	0xFFD06060

Offset: 0x60

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															intf RW 0x1

fpgaintf_en_global Fields

Bit	Name	Description	Access	Reset
0	intf	Used to disable all interfaces between the FPGA and HPS. Software must ensure that all interfaces between the FPGA and HPS are inactive before disabling them. Value: 0 — Disable; 1 — Enable	RW	0x1

fpgaintf_en_0

Used to disable individual interfaces between the FPGA and HPS.
This register is reset only on a cold reset (ignores warm reset).

Module Instance	Base Address	Register Address
i_sys_mgr_core	0xFFD06000	0xFFD06064

Offset: 0x64

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															bscan RW 0x1
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							cfgio RW 0x1	Reserved							rstreq RW 0x1

fpgaintf_en_0 Fields

Bit	Name	Description	Access	Reset
16	bscan	Used to disable the boundary-scan interface. This interface allows the FPGA JTAG TAP controller to execute boundary-scan instructions such as SAMPLE/PRELOAD, EXTEST, and HIGHZ. The boundary-scan interface must be enabled before attempting to send the boundary-scan instructions to the FPGA JTAG TAP controller. **Value** — **Description** 0 — Disable 1 — Enable	RW	0x1
8	cfgio	Used to disable the CONFIG_IO interface. This interface allows the FPGA JTAG TAP controller to execute the CONFIG_IO instruction and configure all device I/Os (FPGA and HPS). This is typically done before executing boundary-scan instructions. The CONFIG_IO interface must be enabled before attempting to send the CONFIG_IO instruction to the FPGA JTAG TAP controller. **Value** — **Description** 0 — Disable 1 — Enable	RW	0x1

Bit	Name	Description	Access	Reset
0	rstreq	Used to disable the reset request interface. This interface allows logic in the FPGA fabric to request HPS resets. This field disables the following reset request signals from the FPGA fabric to HPS:[list] [*]f2s_cold_rst_req - Triggers a cold reset of the HPS[*]f2s_warm_ rst_req - Triggers a warm reset of the HPS[*]f2s_dbg_rst_req - Triggers a debug reset of the HPS[/ list] **Value** **Description** 0 Disable 1 Enable	RW	0x1

fpgaintf_en_1

Used to disable individual interfaces between the FPGA and HPS.
This register is reset only on a cold reset (ignores warm reset).

Module Instance	Base Address	Register Address
i_sys_mgr_core	0xFFD06000	0xFFD06068

Offset: 0x68

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							ctmtrigger RW 0x1	Reserved							stmevent RW 0x1
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							dbgapb RW 0x1	Reserved			trace RW 0x1	Reserved			

fpgaintf_en_1 Fields

Bit	Name	Description	Access	Reset
24	ctmtrigger	Used to disable the FPGA Fabric from sending triggers to HPS debug logic. Note that this doesn't prevent the HPS debug logic from sending triggers to the FPGA Fabric. **Value** — **Description** 0 — Disable 1 — Enable	RW	0x1
16	stmevent	Used to disable the STM event interface. This interface allows logic in the FPGA fabric to trigger events to the STM debug module in the HPS. **Value** — **Description** 0 — Disable 1 — Enable	RW	0x1
8	dbgapb	Used to disable the debug APB interface. This interface allows the HPS debug logic to communicate with debug APB slaves in the FPGA fabric. **Value** — **Description** 0 — Disable 1 — Enable	RW	0x1

Bit	Name	Description	Access	Reset
4	trace	Used to disable the trace interface. This interface allows the HPS debug logic to send trace data to logic in the FPGA fabric. Value: 0 — Disable; 1 — Enable	RW	0x1

fpgaintf_en_2

Used to disable individual interfaces between the FPGA and HPS.
This register is reset only on a cold reset (ignores warm reset).

Module Instance	Base Address	Register Address
i_sys_mgr_core	0xFFD06000	0xFFD0606C

Offset: 0x6C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							spim_1 RW 0x0	Reserved							spim_0 RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							sdmmc RW 0x0	Reserved			nand RW 0x0	Reserved			

fpgaintf_en_2 Fields

Bit	Name	Description	Access	Reset
24	spim_1	Used to disable signals from the FPGA fabric to the SPI master modules that could potentially interfere with their normal operation. The array index corresponds to the SPI master module instance. **Value** — **Description** 0 — Disable 1 — Enable	RW	0x0
16	spim_0	Used to disable signals from the FPGA fabric to the SPI master modules that could potentially interfere with their normal operation. The array index corresponds to the SPI master module instance. **Value** — **Description** 0 — Disable 1 — Enable	RW	0x0
8	sdmmc	Used to disable signals from the FPGA fabric to the SD/MMC controller module that could potentially interfere with its normal operation. **Value** — **Description** 0 — Disable 1 — Enable	RW	0x0
4	nand	Used to disable signals from the FPGA fabric to the NAND flash controller module that could potentially interfere with its normal operation. **Value** — **Description** 0 — Disable 1 — Enable	RW	0x0

fpgaintf_en_3

Used to disable individual interfaces between the FPGA and HPS.
This register is reset only on a cold reset (ignores warm reset).

Module Instance	Base Address	Register Address
i_sys_mgr_core	0xFFD06000	0xFFD06070

Offset: 0x70

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															emac_2 RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							emac_1 RW 0x0	Reserved							emac_0 RW 0x0

fpgaintf_en_3 Fields

Bit	Name	Description	Access	Reset
16	emac_2	Used to disable signals from the FPGA fabric to the EMAC modules that could potentially interfere with their normal operation. The array index corresponds to the EMAC module instance. Value: 0 — Disable; 1 — Enable	RW	0x0

Bit	Name	Description	Access	Reset
8	emac_1	Used to disable signals from the FPGA fabric to the EMAC modules that could potentially interfere with their normal operation. The array index corresponds to the EMAC module instance. Value: Description 0: Disable 1: Enable	RW	0x0
0	emac_0	Used to disable signals from the FPGA fabric to the EMAC modules that could potentially interfere with their normal operation. The array index corresponds to the EMAC module instance. Value: Description 0: Disable 1: Enable	RW	0x0

noc_addr_remap_value

Address remap register. This register drives the remap bits for the NOC.

This is read / write register.

Module Instance	Base Address	Register Address
i_sys_mgr_core	0xFFD06000	0xFFD06080

Offset: 0x80

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														remap1 0x0	remap0 0x0

noc_addr_remap_value Fields

Bit	Name	Description	Access	Reset
1	remap1	if this bit is 0 RAM is NOT available at base address 0x0000_0000 if this bit is 1 RAM is also mapped to base address 0x0000_0000, as long as ROM is not being mapped to the same base address because of remap0 Note: ROM is always available at base address 0xFFFC_0000 RAM is always available at base address 0xFFE0_0000	RW	0x0
0	remap0	if this bit is 0 ROM is also mapped to base address 0x0000_0000 if this bit is 1 ROM is NOT available at base address 0x0000_0000 Note: ROM is always available at base address 0xFFFC_0000 RAM is always available at base address 0xFFE0_0000	RW	0x0

noc_addr_remap_set

This is a Write 1 to Set register.
Writing 0 is ignored, and writing 1 to a specific bit field sets the specific remap bit.
Reads should not return an error, but the actual read value is "Undefined" .

Module Instance	Base Address	Register Address
i_sys_mgr_core	0xFFD06000	0xFFD06084

Offset: 0x84

Access: WO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														remap1 0x0	remap0 0x0

noc_addr_remap_set Fields

Bit	Name	Description	Access	Reset
1	remap1		WO	0x0
0	remap0		WO	0x0

noc_addr_remap_clear

```
        This is a Write 1 to Clear register.
Writing 0 is ignored, and writing 1 to a specific bit field Clears
the specific remap bit.
Reads should not return an error, but the actual read value is
"Undefined" .
```

Module Instance	Base Address	Register Address
i_sys_mgr_core	0xFFD06000	0xFFD06088

Offset: 0x88

Access: WO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														remap1 0x0	remap0 0x0

noc_addr_remap_clear Fields

Bit	Name	Description	Access	Reset
1	remap1		WO	0x0
0	remap0		WO	0x0

ecc_intmask_value

```
          ECC interrupt mask register.
This is a read/write register.
```

Module Instance	Base Address	Register Address
i_sys_mgr_core	0xFFD06000	0xFFD06090

Offset: 0x90

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													ddr1 0x0	ddr0 0x0	sdmmcb 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
sdmmca 0x0	qspi 0x0	nand_rd 0x0	nand_wr 0x0	nand_buf 0x0	dma 0x0	emac2_tx 0x0	emac2_rx 0x0	emac1_tx 0x0	emac1_rx 0x0	emac0_tx 0x0	emac0_rx 0x0	usb1 0x0	usb0 0x0	ocram 0x0	l2 0x0

ecc_intmask_value Fields

Bit	Name	Description	Access	Reset
18	ddr1		RW	0x0
17	ddr0		RW	0x0
16	sdmmcb		RW	0x0
15	sdmmca		RW	0x0
14	qspi		RW	0x0
13	nand_rd		RW	0x0
12	nand_wr		RW	0x0
11	nand_buf		RW	0x0
10	dma		RW	0x0
9	emac2_tx		RW	0x0
8	emac2_rx		RW	0x0
7	emac1_tx		RW	0x0
6	emac1_rx		RW	0x0
5	emac0_tx		RW	0x0
4	emac0_rx		RW	0x0
3	usb1		RW	0x0
2	usb0		RW	0x0
1	ocram		RW	0x0
0	l2		RW	0x0

ecc_intmask_set

```
Write 1 to set a specific modules interrupt mask.
Reads should not return an error, but the actual read value is
"Undefined" .
```

Module Instance	Base Address	Register Address
i_sys_mgr_core	0xFFD06000	0xFFD06094

Offset: 0x94

Access: WO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													ddr1 0x0	ddr0 0x0	sdmmcb 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
sdmmca 0x0	qspi 0x0	nand_rd 0x0	nand_wr 0x0	nand_buf 0x0	dma 0x0	emac2_tx 0x0	emac2_rx 0x0	emac1_tx 0x0	emac1_rx 0x0	emac0_tx 0x0	emac0_rx 0x0	usb1 0x0	usb0 0x0	ocram 0x0	l2 0x0

ecc_intmask_set Fields

Bit	Name	Description	Access	Reset
18	ddr1		WO	0x0
17	ddr0		WO	0x0
16	sdmmcb		WO	0x0
15	sdmmca		WO	0x0
14	qspi		WO	0x0
13	nand_rd		WO	0x0
12	nand_wr		WO	0x0
11	nand_buf		WO	0x0
10	dma		WO	0x0
9	emac2_tx		WO	0x0
8	emac2_rx		WO	0x0
7	emac1_tx		WO	0x0
6	emac1_rx		WO	0x0
5	emac0_tx		WO	0x0
4	emac0_rx		WO	0x0
3	usb1		WO	0x0
2	usb0		WO	0x0
1	ocram		WO	0x0
0	l2		WO	0x0

ecc_intmask_clr

Write 1 to Clear a specific modules interrupt mask.
Reads should not return an error, but the actual read value is "Undefined" .

Module Instance	Base Address	Register Address
i_sys_mgr_core	0xFFD06000	0xFFD06098

Offset: 0x98

Access: WO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													ddr1 0x0	ddr0 0x0	sdmmcb 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
sdmmca 0x0	qspi 0x0	nand_rd 0x0	nand_wr 0x0	nand_buf 0x0	dma 0x0	emac2_tx 0x0	emac2_rx 0x0	emac1_tx 0x0	emac1_rx 0x0	emac0_tx 0x0	emac0_rx 0x0	usb1 0x0	usb0 0x0	ocram 0x0	l2 0x0

ecc_intmask_clr Fields

Bit	Name	Description	Access	Reset
18	ddr1		WO	0x0
17	ddr0		WO	0x0
16	sdmmcb		WO	0x0
15	sdmmca		WO	0x0
14	qspi		WO	0x0
13	nand_rd		WO	0x0
12	nand_wr		WO	0x0
11	nand_buf		WO	0x0
10	dma		WO	0x0
9	emac2_tx		WO	0x0
8	emac2_rx		WO	0x0
7	emac1_tx		WO	0x0
6	emac1_rx		WO	0x0
5	emac0_tx		WO	0x0

Bit	Name	Description	Access	Reset
4	emac0_rx		WO	0x0
3	usb1		WO	0x0
2	usb0		WO	0x0
1	ocram		WO	0x0
0	l2		WO	0x0

ecc_intstatus_serr

ECC single bit error status of individual modules.
A write to this register should return an error.

Module Instance	Base Address	Register Address
i_sys_mgr_core	0xFFD06000	0xFFD0609C

Offset: 0x9C

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													ddr1 0x0	ddr0 0x0	sdmmcb 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
sdmmca 0x0	qspi 0x0	nand_rd 0x0	nand_wr 0x0	nand_buf 0x0	dma 0x0	emac2_tx 0x0	emac2_rx 0x0	emac1_tx 0x0	emac1_rx 0x0	emac0_tx 0x0	emac0_rx 0x0	usb1 0x0	usb0 0x0	ocram 0x0	l2 0x0

ecc_intstatus_serr Fields

Bit	Name	Description	Access	Reset
18	ddr1		RO	0x0
17	ddr0		RO	0x0
16	sdmmcb		RO	0x0
15	sdmmca		RO	0x0
14	qspi		RO	0x0

Bit	Name	Description	Access	Reset
13	nand_rd		RO	0x0
12	nand_wr		RO	0x0
11	nand_buf		RO	0x0
10	dma		RO	0x0
9	emac2_tx		RO	0x0
8	emac2_rx		RO	0x0
7	emac1_tx		RO	0x0
6	emac1_rx		RO	0x0
5	emac0_tx		RO	0x0
4	emac0_rx		RO	0x0
3	usb1		RO	0x0
2	usb0		RO	0x0
1	ocram		RO	0x0
0	l2		RO	0x0

ecc_intstatus_derr

ECC double bit error status of individual modules.
A write to this register should return an error.

Module Instance	Base Address	Register Address
i_sys_mgr_core	0xFFD06000	0xFFD060A0

Offset: 0xA0

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													ddr1 0x0	ddr0 0x0	sdmmcb 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
sdmmca 0x0	qspi 0x0	nand_rd 0x0	nand_wr 0x0	nand_buf 0x0	dma 0x0	emac2_tx 0x0	emac2_rx 0x0	emac1_tx 0x0	emac1_rx 0x0	emac0_tx 0x0	emac0_rx 0x0	usb1 0x0	usb0 0x0	ocram 0x0	l2 0x0

ecc_intstatus_derr Fields

Bit	Name	Description	Access	Reset
18	ddr1		RO	0x0
17	ddr0		RO	0x0
16	sdmmcb		RO	0x0
15	sdmmca		RO	0x0
14	qspi		RO	0x0
13	nand_rd		RO	0x0
12	nand_wr		RO	0x0
11	nand_buf		RO	0x0
10	dma		RO	0x0
9	emac2_tx		RO	0x0
8	emac2_rx		RO	0x0
7	emac1_tx		RO	0x0
6	emac1_rx		RO	0x0
5	emac0_tx		RO	0x0
4	emac0_rx		RO	0x0
3	usb1		RO	0x0
2	usb0		RO	0x0
1	ocram		RO	0x0
0	l2		RO	0x0

mpu_status_l2_ecc

This is a read only register which reads the current mpu L2 ecc interrupt status.
A write to this register should return an error.

Module Instance	Base Address	Register Address
i_sys_mgr_core	0xFFD06000	0xFFD060A4

Offset: 0xA4

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
merr_pending 0x0	Reserved			merr_info 0x0											
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
serr_pending 0x0	Reserved			serr_info 0x0											

mpu_status_l2_ecc Fields

Bit	Name	Description	Access	Reset
31	merr_pending	Unmaksed value of a pending multiple bits ECC error status.	RO	0x0
27:16	merr_info	12 bit Serr Info field. In Baum this will be the index and way information where the ECC error occured.	RO	0x0
15	serr_pending	Unmaksed value of a pending single bit ECC error status.	RO	0x0
11:0	serr_info	12 bit Serr Info field. In Baum this will be the index and way information where the ECC error occured.	RO	0x0

mpu_clear_l2_ecc

Write 1 to Clear register to clear the specific bit field of mpu l2 ecc interrupt pending status
Reads should not return an error, but the read value is undefined.

Module Instance	Base Address	Register Address
i_sys_mgr_core	0xFFD06000	0xFFD060A8

Offset: 0xA8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
merr 0x0	Reserved														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
serr 0x0	Reserved														

mpu_clear_l2_ecc Fields

Bit	Name	Description	Access	Reset
31	merr	Write 1 to this field to clear the MPU L2 ECC multiple bit Error interrupt Status and the actual Interrupt.	RW	0x0
15	serr	Write 1 to this field to clear the MPU L2 ECC single bit Error interrupt Status and the actual Interrupt.	RW	0x0

mpu_status_l1_parity

```
          Parity status from L1 and scu. This is a read only register.
A write to this register should return an error.

[17] CPU1 SCU parity error
[16] CPU0 SCU parity error

[15] CPU1 BTAC parity error
[14] CPU1 GHB parity error
[13] CPU1 instruction tag RAM parity error
[12] CPU1 instruction data RAM parity error
[11] CPU1 main TLB parity error
[10] CPU1 data outer RAM parity error
```

```
[9]  CPU1 data tag RAM parity error
[8]  CPU1 data data RAM parity error.

[7]  CPU0 BTAC parity error
[6]  CPU0 GHB parity error
[5]  CPU0 instruction tag RAM parity error
[4]  CPU0 instruction data RAM parity error
[3]  CPU0 main TLB parity error
[2]  CPU0 data outer RAM parity error
[1]  CPU0 data tag RAM parity error
[0]  CPU0 data data RAM parity error.
```

Module Instance	Base Address	Register Address
i_sys_mgr_core	0xFFD06000	0xFFD060AC

Offset: 0xAC

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved														scu 0x0	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cpu1 0x0								cpu0 0x0							

mpu_status_l1_parity Fields

Bit	Name	Description	Access	Reset
17:16	scu	SCU parity interrupt status	RO	0x0
15:8	cpu1	CPU1 L1 parity interrupt status	RO	0x0
7:0	cpu0	CPU0 L1 parity interrupt status	RO	0x0

mpu_clear_l1_parity

Parity status clear bit.

A write to 1 of a specific bit clears the curresponding parity status bit.
A read of this register should not return an error, but the actual read value is undefined.

[17] CPU1 SCU parity error
[16] CPU0 SCU parity error

[15] CPU1 BTAC parity error
[14] CPU1 GHB parity error
[13] CPU1 instruction tag RAM parity error
[12] CPU1 instruction data RAM parity error
[11] CPU1 main TLB parity error
[10] CPU1 data outer RAM parity error
[9] CPU1 data tag RAM parity error
[8] CPU1 data data RAM parity error.

[7] CPU0 BTAC parity error
[6] CPU0 GHB parity error
[5] CPU0 instruction tag RAM parity error
[4] CPU0 instruction data RAM parity error
[3] CPU0 main TLB parity error
[2] CPU0 data outer RAM parity error
[1] CPU0 data tag RAM parity error
[0] CPU0 data data RAM parity error.

Module Instance	Base Address	Register Address
i_sys_mgr_core	0xFFD06000	0xFFD060B0

Offset: 0xB0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved														scu 0x0	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cpu1 0x0								cpu0 0x0							

mpu_clear_l1_parity Fields

Bit	Name	Description	Access	Reset
17:16	scu	SCU parity interrupt clear. Write 1 to Clear	RW	0x0
15:8	cpu1	CPU1 L1 parity interrupt clear. Write 1 to Clear	RW	0x0
7:0	cpu0	CPU0 L1 parity interrupt clear. Write 1 to Clear	RW	0x0

mpu_set_l1_parity

```
          Parity status set bit.
A write to 1 of a specific bit sets the curresponding parity status
bit.
This register is used only to check the specific ISR routine.
A read of this register should not return an error, but the actual
read value is undefined.

[17] CPU1 SCU parity error
[16] CPU0 SCU parity error

[15] CPU1 BTAC parity error
[14] CPU1 GHB parity error
[13] CPU1 instruction tag RAM parity error
[12] CPU1 instruction data RAM parity error
[11] CPU1 main TLB parity error
[10] CPU1 data outer RAM parity error
[9]  CPU1 data tag RAM parity error
[8]  CPU1 data data RAM parity error.

[7]  CPU0 BTAC parity error
[6]  CPU0 GHB parity error
[5]  CPU0 instruction tag RAM parity error
[4]  CPU0 instruction data RAM parity error
[3]  CPU0 main TLB parity error
[2]  CPU0 data outer RAM parity error
[1]  CPU0 data tag RAM parity error
[0]  CPU0 data data RAM parity error.
```

Module Instance	Base Address	Register Address
i_sys_mgr_core	0xFFD06000	0xFFD060B4

Offset: 0xB4

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved														scu 0x0	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cpu1 0x0								cpu0 0x0							

mpu_set_l1_parity Fields

Bit	Name	Description	Access	Reset
17:16	scu	SCU parity interrupt set. Write 1 to Set	RW	0x0
15:8	cpu1	CPU1 L1 parity interrupt set. Write 1 to Set	RW	0x0
7:0	cpu0	CPU0 L1 parity interrupt set. Write 1 to Set	RW	0x0

noc_timeout

Module Instance	Base Address	Register Address
i_sys_mgr_core	0xFFD06000	0xFFD060C0

Offset: 0xC0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															en 0x0

noc_timeout Fields

Bit	Name	Description	Access	Reset
0	en	NOC Timeout Enable. You can set this bit to enable an automatic timeout that flushes all pending transactions in the HPS-to-FPGA and lightweight HPS-to-FPGA bridges when they are shutdown and reset. To use this automatic timeout, the bridges must be operating above 100 MHz. The automatic timeout duration is approximately 320 ns. If your system application handles and completes all transactions before shutting down the bridges, this bit does not need to be enabled.	RW	0x0

noc_idlereq_set

Set IDLE request to each NOC master.

Module Instance	Base Address	Register Address
i_sys_mgr_core	0xFFD06000	0xFFD060C4

Offset: 0xC4

Access: WO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							fpga2sdram2 0x0	Reserved			fpga2sdram1 0x0	Reserved			fpga2sdram0 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							fpga2soc 0x0	Reserved			lwsoc2fpga 0x0	Reserved			soc2fpga 0x0

noc_idlereq_set Fields

Bit	Name	Description	Access	Reset
24	fpga2sdram2		WO	0x0
20	fpga2sdram1		WO	0x0
16	fpga2sdram0		WO	0x0
8	fpga2soc		WO	0x0
4	lwsoc2fpga		WO	0x0
0	soc2fpga		WO	0x0

noc_idlereq_clr

Clear IDLE request to each NOC master.

Module Instance	Base Address	Register Address
i_sys_mgr_core	0xFFD06000	0xFFD060C8

Offset: 0xC8

Access: WO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							fpga2sdram2 0x0	Reserved			fpga2sdram1 0x0	Reserved			fpga2sdram0 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							fpga2soc 0x0	Reserved			lwsoc2fpga 0x0	Reserved			soc2fpga 0x0

noc_idlereq_clr Fields

Bit	Name	Description	Access	Reset
24	fpga2sdram2		WO	0x0
20	fpga2sdram1		WO	0x0
16	fpga2sdram0		WO	0x0
8	fpga2soc		WO	0x0
4	lwsoc2fpga		WO	0x0
0	soc2fpga		WO	0x0

noc_idlereq_value

```
IDLE request to each NOC master.

This register can be set by writing 1 to the specific bit in
noc_idlereq_set register.
This register can be cleared by writing 1 to the specific bit in
noc_idlereq_clr register
```

Module Instance	Base Address	Register Address
i_sys_mgr_core	0xFFD06000	0xFFD060CC

Offset: 0xCC

Access: 'not specified'

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							fpga2sdram2 0x0	Reserved			fpga2sdram1 0x0	Reserved			fpga2sdram0 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							fpga2soc 0x0	Reserved			lwsoc2fpga 0x0	Reserved			soc2fpga 0x0

noc_idlereq_value Fields

Bit	Name	Description	Access	Reset
24	fpga2sdram2		not specified	0x0
20	fpga2sdram1		not specified	0x0
16	fpga2sdram0		not specified	0x0
8	fpga2soc		not specified	0x0
4	lwsoc2fpga		not specified	0x0
0	soc2fpga		not specified	0x0

noc_idleack

Idle acknowledge value from NOC Masters. This is asserted (value 1 in the field) in response to the IDLE requests asserted by software.

Module Instance	Base Address	Register Address
i_sys_mgr_core	0xFFD06000	0xFFD060D0

Offset: 0xD0

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							fpga2sdram2 0x0	Reserved			fpga2sdram1 0x0	Reserved			fpga2sdram0 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							fpga2soc 0x0	Reserved			lwsoc2fpga 0x0	Reserved			soc2fpga 0x0

noc_idleack Fields

Bit	Name	Description	Access	Reset
24	fpga2sdram2		RO	0x0
20	fpga2sdram1		RO	0x0
16	fpga2sdram0		RO	0x0
8	fpga2soc		RO	0x0
4	lwsoc2fpga		RO	0x0
0	soc2fpga		RO	0x0

noc_idlestatus

Status of IDLE from the NOC masters. A 1 in the field means the specific master is idle.

Module Instance	Base Address	Register Address
i_sys_mgr_core	0xFFD06000	0xFFD060D4

Offset: 0xD4

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							fpga2sdram2 0x0	Reserved			fpga2sdram1 0x0	Reserved			fpga2sdram0 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							fpga2soc 0x0	Reserved			lwsoc2fpga 0x0	Reserved			soc2fpga 0x0

noc_idlestatus Fields

Bit	Name	Description	Access	Reset
24	fpga2sdram2		RO	0x0
20	fpga2sdram1		RO	0x0
16	fpga2sdram0		RO	0x0
8	fpga2soc		RO	0x0
4	lwsoc2fpga		RO	0x0
0	soc2fpga		RO	0x0

fpga2soc_ctrl

Module Instance	Base Address	Register Address
i_sys_mgr_core	0xFFD06000	0xFFD060D8

Offset: 0xD8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															allow_secure 0x0

fpga2soc_ctrl Fields

Bit	Name	Description	Access	Reset
0	allow_secure	0 - All Transactions from FPGA2SOC is converted to be Non-Secure 1 - Both Secure and Non-Secure Transactions is allowed by FPGA2SOC.	RW	0x0

sys_mgr_rom Address Map

Module Instance	Base Address	End Address
i_sys_mgr_rom	0xFFD06200	0xFFD06FFF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
romhw_ctrl on page 5-94	0x0	32	RW	0x100	Boot ROM Hardware Control Register
romcode_ctrl on page 5-95	0x4	32	RW	0x0	Control Register
romcode_qspi_reset_command on page 5-96	0x8	32	RW	0x0	QSPI reset command
romcode_initswstate on page 5-97	0xC	32	RW	0x0	Initial Software State Register
romcode_initswlastld on page 5-98	0x10	32	RW	0x0	Initial Software Last Image Loaded Register

Register	Offset	Width	Access	Reset Value	Description
warmram_enable on page 5-99	0x18	32	RW	0x0	Enable Register
warmram_datastart on page 5-100	0x1C	32	RW	0x0	Data Start Register
warmram_length on page 5-101	0x20	32	RW	0x0	Length Register
warmram_execution on page 5-102	0x24	32	RW	0x0	Execution Register
warmram_crc on page 5-103	0x28	32	RW	0x0	Expected CRC Register
isw_handoff on page 5-104	0x30	32	RW	0x0	Preloader to OS Handoff Information
romcode_bootromsw-state on page 5-105	0x50	32	RW	0x0	Preloader to OS Handoff Information
romcode_stickyset_warmclr on page 5-106	0x70	32	'not specified'	0x0	Write 1 to set each bit. Write 0 has no effect. Clears on warm/cold reset. No other way to clear the value once written 1.
romcode_stickyset_coldclr on page 5-106	0x74	32	'not specified'	0x0	Write 1 to set each bit. Write 0 has no effect. Clears on cold reset. No other way to clear the value once written 1.

sys_mgr_rom Summary

Base Address: 0xFFD06200

Register Address Offset	Bit Fields
i_sys_mgr_rom	

Register Address Offset	Bit Fields															
romhw_ctrl 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															waitstate RW 0x0
romcode_ctrl 0x4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														warmrstcfgio RW 0x0	warmrstcfgpinmux RW 0x0
romcode_qspi_reset_command 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	value RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	value RW 0x0															
romcode_initswstate 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	value RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	value RW 0x0															
romcode_initswlastld 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														index RW 0x0	
warmram_enable 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	magic RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	magic RW 0x0															

Register Address Offset	Bit Fields															
warmram_datastart 0x1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	offset RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	offset RW 0x0															
warmram_length 0x20	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	size RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	size RW 0x0															
warmram_execution 0x24	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	offset RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	offset RW 0x0															
warmram_crc 0x28	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	expected RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	expected RW 0x0															
isw_handoff 0x30	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	isw_handoff RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	isw_handoff RW 0x0															
romcode_bootromswstate 0x50	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	value RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	value RW 0x0															

Register Address Offset	Bit Fields															
romcode_stickyset_warmclr 0x70	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	value 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	value 0x0															
romcode_stickyset_coldclr 0x74	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	value 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	value 0x0															

romhw_ctrl

Controls behavior of Boot ROM hardware.
Reset only on a cold reset.

Module Instance	Base Address	Register Address
i_sys_mgr_rom	0xFFD06200	0xFFD06200 - 0xFFD0621C

Offset: 0x0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															waitstate RW 0x0

romhw_ctrl Fields

Bit	Name	Description	Access	Reset
0	waitstate	Controls the number of wait states applied to the Boot ROM's read operation. This field is cleared on a cold reset and optionally updated by hardware upon deassertion of warm reset. **Value** — **Description** 0 — Disable 1 — Enable	RW	0x0

romcode_ctrl

Contains information used to control Boot ROM code. Reset only on a cold reset.

Module Instance	Base Address	Register Address
i_sys_mgr_rom	0xFFD06200	0xFFD06204 - 0xFFD06220

Offset: 0x4

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														warmrstcfgio RW 0x0	warmrstcfgpinmux RW 0x0

romcode_ctrl Fields

Bit	Name	Description	Access	Reset
1	warmrstcfgio	Specifies whether the Boot ROM code configures the IOs used by boot after a warm reset. Note that the Boot ROM code always configures the IOs used by boot after a cold reset. After the Boot ROM code configures the IOs used by boot, it always disables this field. It is up to user software to enable this field if it wants a different behavior. **Value** — **Description** 0 — disabled 1 — enabled	RW	0x0
0	warmrstcfgpinmux	Specifies whether the Boot ROM code configures the pin mux for boot pins after a warm reset. Note that the Boot ROM code always configures the pin mux for boot pins after a cold reset. After the Boot ROM code configures the pin mux for boot pins, it always disables this field. It is up to user software to enable this field if it wants a different behavior. **Value** — **Description** 0 — disabled 1 — enabled	RW	0x0

romcode_qspi_reset_command

Information used by bootrom to send specific reset command sequences to QSPI.
This register value gets reset only on a cold reset.

Below are the definisions used by bootrom
0xXXXXXX00: Don't software reset.
0xXXXXXX01: Send 0x66, 0x99 as reset command.
0xXXXXXX02: Send 0xFF as reset command.
0xXXXXXX03: Send TBA as reset command.

There is also a custom command for devices we don't know about yet.

```
0xXXXXZZ81: Send byte 0xZZ as reset command.
0xXXWWZZ82: Send byte 0xZZ, 0xWW as reset command.
0xYYWWZZ83: Send byte 0xZZ, 0xWW, 0xYY as reset command.
```

Module Instance	Base Address	Register Address
i_sys_mgr_rom	0xFFD06200	0xFFD06208 - 0xFFD06224

Offset: 0x8

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
value RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
value RW 0x0															

romcode_qspi_reset_command Fields

Bit	Name	Description	Access	Reset
31:0	value	Address for CPU1 to start executing at after coming out of reset.	RW	0x0

romcode_initswstate

```
        Initial Software loaded by the Boot ROM writes the magic
value 0x49535756 (ISWV in ASCII) to this register when it has reached
a valid state.
Reset only on a cold reset.
```

Module Instance	Base Address	Register Address
i_sys_mgr_rom	0xFFD06200	0xFFD0620C - 0xFFD06228

Offset: 0xC

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
value RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
value RW 0x0															

romcode_initswstate Fields

Bit	Name	Description	Access	Reset
31:0	value	Written with magic value. Value: Description 0: Invalid 1230198614: Valid	RW	0x0

romcode_initswlastld

Contains the index of the last Initial Software image loaded by the Boot ROM from the boot device.
Reset only on a cold reset.

Module Instance	Base Address	Register Address
i_sys_mgr_rom	0xFFD06200	0xFFD06210 - 0xFFD0622C

Offset: 0x10

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														index RW 0x0	

romcode_initswlastld Fields

Bit	Name	Description	Access	Reset
1:0	index	Index of last image loaded.	RW	0x0

warmram_enable

Enables or disables the warm reset from On-chip RAM feature. Reset only on a cold reset.

Module Instance	Base Address	Register Address
i_sys_mgr_rom	0xFFD06200	0xFFD06218 - 0xFFD06234

Offset: 0x18

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
magic RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
magic RW 0x0															

warmram_enable Fields

Bit	Name	Description	Access	Reset
31:0	magic	Controls whether Boot ROM will attempt to boot from the contents of the On-chip RAM on a warm reset. When this feature is enabled, the Boot ROM code will not configure boot IOs, the pin mux, or clocks. Note that the enable value is a 32-bit magic value (provided by the enum). **Value** — **Description** 0x0 — Boot ROM code will not attempt to boot from On-chip RAM on a warm reset 0xae9efebc — Boot ROM code will attempt to boot from On-chip RAM on a warm reset	RW	0x0

warmram_datastart

Offset into On-chip RAM of the start of the region for CRC validation
Reset only on a cold reset.

Module Instance	Base Address	Register Address
i_sys_mgr_rom	0xFFD06200	0xFFD0621C - 0xFFD06238

Offset: 0x1C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
offset RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
offset RW 0x0															

warmram_datastart Fields

Bit	Name	Description	Access	Reset
31:0	offset	Contains the byte offset into the On-chip RAM of the start of the On-chip RAM region for the warm boot CRC validation. The offset must be an integer multiple of 4 (i.e. aligned to a word). The Boot ROM code will set the top 16 bits to 0xFFFF and clear the bottom 2 bits.	RW	0x0

warmram_length

Length of region in On-chip RAM for CRC validation. Reset only on a cold reset.

Module Instance	Base Address	Register Address
i_sys_mgr_rom	0xFFD06200	0xFFD06220 - 0xFFD0623C

Offset: 0x20

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
size RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
size RW 0x0															

warmram_length Fields

Bit	Name	Description	Access	Reset
31:0	size	Contains the length (in bytes) of the region in the On-chip RAM for the warm boot CRC validation. If the length is 0, the Boot ROM won't perform CRC calculation and CRC check to avoid overhead caused by CRC validation. If the START + LENGTH exceeds the maximum offset into the On-chip RAM, the Boot ROM won't boot from the On-chip RAM. The length must be an integer multiple of 4. The Boot ROM code will clear the top 16 bits and the bottom 2 bits.	RW	0x0

warmram_execution

Offset into On-chip RAM to enter to on a warm boot. Reset only on a cold reset.

Module Instance	Base Address	Register Address
i_sys_mgr_rom	0xFFD06200	0xFFD06224 - 0xFFD06240

Offset: 0x24

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
offset RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
offset RW 0x0															

warmram_execution Fields

Bit	Name	Description	Access	Reset
31:0	offset	Contains the global address into the On-chip RAM that the Boot ROM jumps to if the CRC validation succeeds. This value is a logical address, not an offset from a base address.	RW	0x0

warmram_crc

Length of region in On-chip RAM for CRC validation. Reset only on a cold reset.

Module Instance	Base Address	Register Address
i_sys_mgr_rom	0xFFD06200	0xFFD06228 - 0xFFD06244

Offset: 0x28

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
expected RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
expected RW 0x0															

warmram_crc Fields

Bit	Name	Description	Access	Reset
31:0	expected	Contains the expected CRC of the region in the On-chip RAM.The Boot ROM code calculates the actual CRC for all bytes in the region specified by the DATA START an LENGTH registers. The contents of the EXECUTION register (after it has been read and modified by the Boot ROM code) is also included in the CRC calculation. The contents of the EXECUTION register is added to the CRC accumulator a byte at a time starting with the least significant byte. If the actual CRC doesn't match the expected CRC value in this register, the Boot ROM won't boot from the On-chip RAM. The CRC is a standard CRC32 with the polynomial: x^32 + x^26 + x^23 + x^22 + x^16 + x^12 + x^11 + x^10 + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1 There is no reflection of the bits and the initial value of the remainder is 0xFFFFFFFF and the final value is exclusive ORed with 0xFFFFFFFF.	RW	0x0

isw_handoff

These registers are used to store handoff infomation between the preloader and the OS. These 8 registers can be used to store any information. The contents of these registers have no impact on the state of the HPS hardware.
A total of 8 x 32 bit registers for Second State Boot Loader handoff
Reset only on a cold reset.

Module Instance	Base Address	Register Address
i_sys_mgr_rom	0xFFD06200	0xFFD06230 - 0xFFD0624C

Offset: 0x30 - 0x4C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
isw_handoff RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
isw_handoff RW 0x0															

isw_handoff Fields

Bit	Name	Description	Access	Reset
31:0	isw_handoff		RW	0x0

romcode_bootromswstate

general purpose register used by the Boot ROM code. Actual usage is defined in the Boot ROM source code.
Reset only on a cold reset.

Module Instance	Base Address	Register Address
i_sys_mgr_rom	0xFFD06200	0xFFD06250 - 0xFFD0626C

Offset: 0x50 - 0x6C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
value RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
value RW 0x0															

romcode_bootromswstate Fields

Bit	Name	Description	Access	Reset
31:0	value		RW	0x0

romcode_stickyset_warmclr

```
        Write 1 to set each bit.
Write 0 has no effect.
Clears on warm/cold reset. No other way to clear the value once
written 1.
```

Module Instance	Base Address	Register Address
i_sys_mgr_rom	0xFFD06200	0xFFD06270 - 0xFFD0628C

Offset: 0x70

Access: 'not specified'

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
value 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
value 0x0															

romcode_stickyset_warmclr Fields

Bit	Name	Description	Access	Reset
31:0	value		not specified	0x0

romcode_stickyset_coldclr

```
        Write 1 to set each bit.
Write 0 has no effect.
Clears on cold reset. No other way to clear the value once written 1.
```

Module Instance	Base Address	Register Address
i_sys_mgr_rom	0xFFD06200	0xFFD06274 - 0xFFD06290

Offset: 0x74

Access: 'not specified'

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
value 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
value 0x0															

romcode_stickyset_coldclr Fields

Bit	Name	Description	Access	Reset
31:0	value		not specified	0x0

Document Revision History

Table 5-2: Document Revision History

Date	Version	Changes
October 2016	2016.10.28	Maintenance release.
May 2016	2016.05.27	Removed the following references to the Ethernet application interface: • app_clk_sel content in the *EMAC* section • emac_*_switch bits in the fpgaintf_en_3 register • app_clk_sel bits in the emac* registers The Ethernet application interface (also called the switch interface) is not supported.
May 2016	2016.05.03	Maintenance release.
November 2015	2015.11.02	Maintenance release.
May 2015	2015.05.04	Maintenance release.

Date	Version	Changes
December 2014	2014.12.15	• Block Diagram updated: Slave port defined • "NAND Flash Controller" section updated • "USB 2.0 OTG" section updated • "EMAC" section updated
August 2014	2014.08.18	Initial release.

SoC Security 6

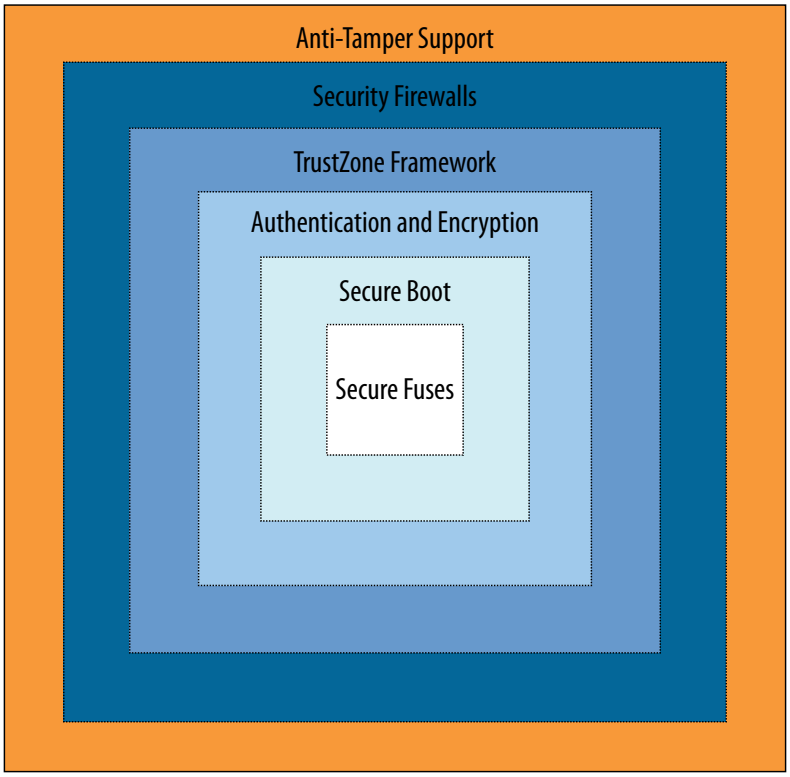
2015.10.30

a10_5v4 Subscribe Send Feedback

Note: For a complete NDA version of the entire Security SoC chapter, address map and register definitions, contact your local field sales office.

The Arria 10 SoC provides the framework for implementing secure systems by offering a layered hardware and software solution. A dedicated Security Manager module in the Hard Processor System (HPS) supervises secure initialization of the SoC and manages system response during tamper events. User fuses offer secure boot options and registers within the Security Manager provide further enhancement of secure states. Using Elliptical Curve Digital Signal Algorithm (ECDSA256) with Secure Hash Algorithm (SHA256) and Advanced Encryption Standard (AES) algorithms in the FPGA's Configuration Subsystem (CSS), boot images can be authenticated and decrypted. The integration of the ARM TrustZone® technology and security firewalls within the Arria 10 system interconnect provides defined partitioning in the device for secure and non-secure accesses. Protection mechanisms are integrated into debug components of the SoC to allow varying levels of visibility for debug.

Figure 6-1: Arria 10 Layered Security Solution



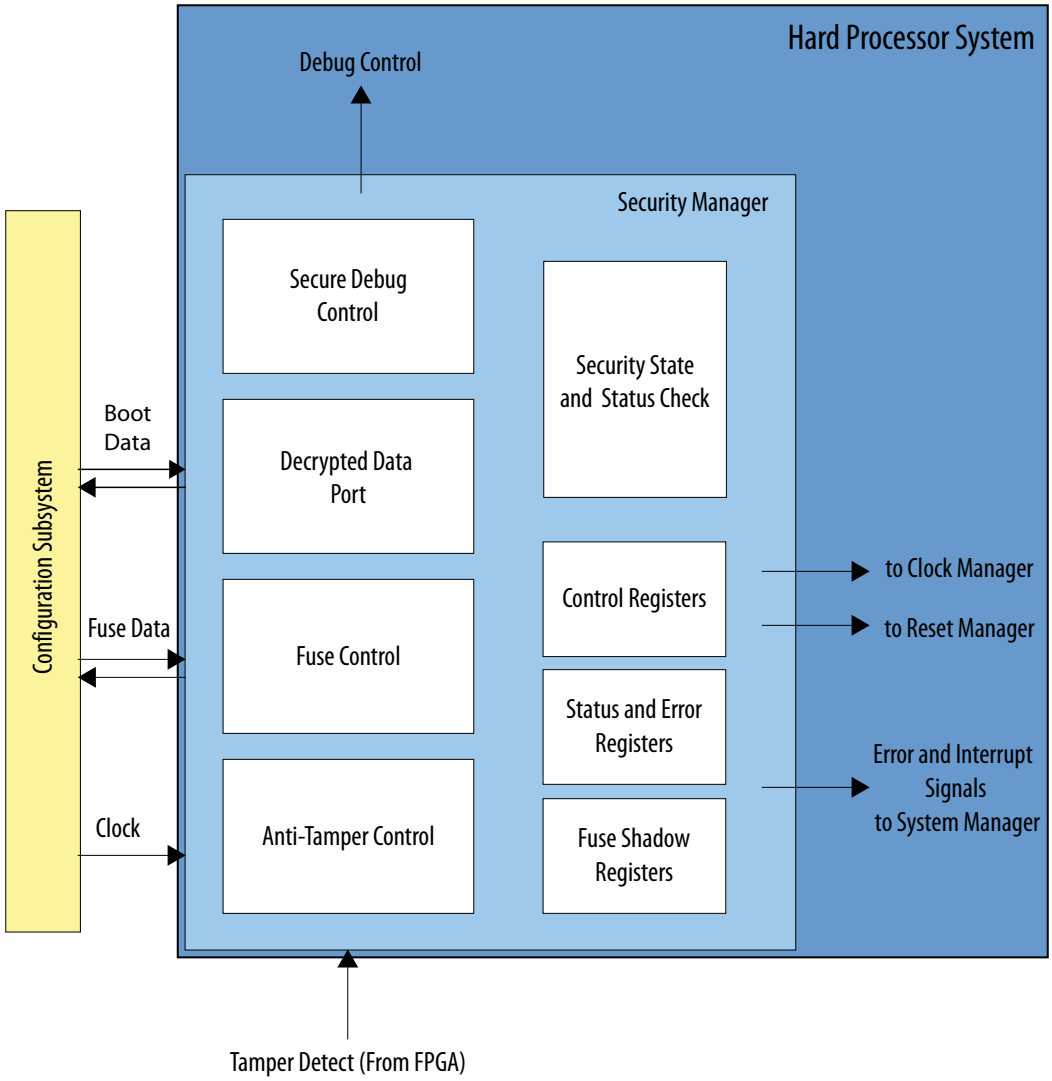
Security Manager

The Security Manager module is integrated in the Hard Processor System (HPS) and provides the overall management of security within the SoC, including:

- Recognition of secure fuse configuration
- Secure state control and status check of security features
- Secure boot options
- Varying levels of debug visibility
- Anti-Tamper support

Security Manager Block Diagram

Figure 6-2: Security Manager Block Diagram



Functional Overview

The Security Manager integrates several functions that support the TrustZone technology, manage security states in the device, and hold secure fuse information.

The main components of the Security Manager architecture include:

- Fuse Control:

  When the HPS is powered, the Security Manager ensures reliable and verified receipt of the fuse information from the Configuration Subsystem (CSS) in the FPGA, stores it in fuse shadow registers and can request further fuse information.
- Security State and Status Check:

  This sub-module holds the security state of the system, which is controlled by the fuse bits, hardware and software programming. This sub-module also has the ability to check and raise the level of security.
- Encryption Data Port:

  This interface receives authenticated and decrypted boot images from the CSS.

  - Support for ECDSA256 (SHA256) authenticated boot.
  - Support for AES-based encrypted boot.
- Registers:

  - Control Registers configure security state and debug options for the device.
  - Status and Error Registers flag transmission errors and interrupts.
  - Fuse Shadow Registers hold a copy of the user fuse information.
- Anti-Tamper Control:

  On a tamper event, this module sends a signal to the Reset Manager to initiate the scrambling and clearing of all memories, including on-chip RAM, peripheral memories, L1 cache and L2 cache. Upon completion, the Security Manager sends a signal to the FPGA to indicate that the anti-tamper event has been handled.

Functional Description

Secure Initialization Overview

Secure initialization allows the device to boot and configure in a secured state.

Secure initialization begins with the CSS module. On FPGA power-up, the Configuration Subsystem (CSS) powers, initializes and loads the fuse bits. The CSS sends the FPGA its fuse configuration information. If the HPS is powered, the CSS sends the HPS fuse information to the Security Manager. This information is held in the `HPS_fusesec` shadow register in the Security Manager.

When the Security Manager is released from reset, it requests configuration information from the CSS and performs security checks. At this point, the rest of the HPS is still in reset. The security checks validate whether the state of each security option is valid. The Security Manager decodes the fuse bits and brings the rest of the HPS out of reset.

If there are errors in the initial transmission of the secure fuse information, the HPS stays in reset and no automatic retry occurs. Only an FPGA re-configuration causes a retry.

When the HPS is released from reset, the Security Manager sends signals to initialize the system blocks, such as the Clock Manager, FPGA Manager, System Manager. The clock control fuse information is

automatically sent to the Clock Manager, the memory control fuse information is automatically sent to the Reset Manager and all other fuse functions (authentication, encryption, and public key source and length) are stored in a memory-mapped location for the boot ROM code to read. After these tasks are successfully completed, CPU0 comes out of reset in a secure state.

After CPU0 is released from reset, the boot ROM begins executing. At this time, the HPS is in a trusted state and the boot ROM code is guaranteed to execute as expected. For both secure and non-secure boot, all slave peripherals are brought out of reset in a secure state.

After initial security controls have been set, the boot ROM determines the second-stage boot loader source from the `bootinfo` register in the System Manager. The boot source can be from external memory through the HPS or through the FPGA. If the second-stage boot loader code comes from HPS external memory, then the boot ROM configures the clock and the interface to external memory.

When the boot ROM attempts to read from the flash, it looks for one of four images in external memory. Initially, it looks for image 0. If it is found, the image is authenticated, decrypted, or both depending on the requirements of the current security state. If these steps are successful, then the image is executed. However, if any of these steps fail, then the boot ROM moves onto the next image until it runs out of images.

For example, during a secure boot, the second-stage boot loader header includes an authentication key and the incoming image is authenticated. If indicated from the current security state, the image may be decrypted as well. The boot ROM contains functions to establish transitive trust to the second-stage boot code loaded from external flash or from the FPGA into on-chip RAM. If trust cannot be established in a secure boot, the HPS does not come out of reset and can attempt to load a different second-stage boot image.

For comprehensive information on the booting and boot image partitioning refer to the *Booting and Configuration* appendix of the *Arria 10 Hard Processor Technical Reference Manual*.

If all the images fail, it attempts to locate an image from the FPGA fabric and boot from there. If the FPGA boot fails, then it halts.

If the HPS receives its second-stage boot loader code from the FPGA, it waits for the FPGA to finish configuration before it obtains the image from FPGA RAM.

If required, a portion of the second-stage boot loader header can be used to raise security states of security features in the device that are currently in non-secure mode.

Related Information

- **Secure Boot** on page 6-15
  For more information about the secure boot process, refer to the "Secure Boot" section.
- **Booting and Configuration** on page 30-1
  For more information about the boot process refer to the *Booting and Configuration* appendix.

Secure Fuses

During initialization of the device, the Configuration Subsystem (CSS) sends the value of the HPS secure fuses to the HPS and holds a copy of the FPGA secure fuses. The HPS and FPGA each hold their own fuse values. However, a copy of these fuse values is held in the `HPS_fusesec` and `fpga_fusesec` registers within the Security Manager.

The following table details the HPS security fuse bits sent by the CSS to the Security Manager and contained within the HPS_fusesec register. A "blown" fuse state is represented by a 1 in the HPS_fusesec register and a "not blown" fuse state is represented by a 0.

Table 6-1: HPS_fusesec Register Description

Bits	Name	Description
31:27	Reserved	Bit values in this field are undefined.
26:23	csel_f	This field indicates the value of the clock select fuses that are available for configuring the clock for the boot interface and for the PLLs. Refer to the *Clock Configuration* section for more information on CSEL encodings.
22	dbg_access_f	This fuse determines the initial state of the debug access domains.
21	dbg_lock_JTAG	This field indicates if the HPS JTAG access level can be changed through software when the HPS is released from reset. • 0x0= HPS JTAG access level can be changed through the sec_jtagdbg register. • 0x1= HPS JTAG access level cannot be changed (locked).
20	dbg_lock_DAP	This field indicates if the DAP access level can be changed through software when the HPS is released from reset. • 0x0= The DAP access level can be changed through the sec_dapdbg register. • 0x1= The DAP access level cannot be changed (locked).
19	dbg_lock_CPU0	This field indicates if the CPU0 debug access level can be changed through software when the HPS is released from reset. • 0x0= CPU0 debug access level can be changed through the sec_cpu0dbg register. • 0x1= CPU0 debug access level cannot be changed (locked).
18	dbg_lock_CPU1	This field indicates if the CPU1 debug access level can be changed through software when the HPS is released from reset. • 0x0= The CPU1 debug access level can be changed through the sec_cpu1dbg register. • 0x1= The CPU1 debug access level cannot be changed (locked).

Bits	Name	Description
17	dbg_lock_CS	This field indicates if the CoreSight debug access level can be changed through software when the HPS is released from reset. • 0x0= The CoreSight debug access level can be changed through the sec_csdbg register. • 0x1= The CoreSight debug access level cannot be changed (locked).
16	dbg_lock_FPGA	This field indicates if the FPGA debug access level can be changed through software when the HPS is released from reset. • 0x0= The FPGA debug access level can be changed through the sec_fpgadbg register. • 0x1= The FPGA debug access level cannot be changed (locked).
15:12	Reserved	Bit values in this field are undefined.
11	clr_ram_order_f	This fuse value determines how RAMs are cleared during a tamper event. • 0x0= All RAMs are cleared in parallel. • 0x1= All RAMs are cleared in series.
10	clr_ram_cold_f	This fuse value indicates what happens to the RAM on a cold reset. • 0x0= All RAMs are not cleared on a cold reset. • 0x1= All RAMs are cleared on a cold reset.
9	clr_ram_warm_f	This fuse value indicates what happens to the RAMs on a warm reset. • 0x0= All RAMs are not cleared on a warm reset. • 0x1= All RAMs are cleared on a warm reset.
8	oc_boot_f	This fuse value determines if the second-stage boot code is allowed to boot from on-chip RAM. • 0x0= Second-stage boot can be from on-chip RAM if enabled by the System Manager. • 0x1= Second-stage boot is not from on-chip RAM.

Bits	Name	Description
7	hps_clk_f	This fuse value selects the clock used for the boot process and in the case of a tamper event, memory scrambling. • 0x0= The external oscillator, EOSC1, is used for boot. • 0x1= The internal oscillator, cb_intosc_ls_clk, is used for boot.
6	fpga_boot_f	If blown, this fuse value allows the FPGA to configure independently and allows the HPS to boot from an encrypted next-stage boot source that was decrypted into the FPGA. • 0x0= Booting is dependent on the BSEL pins. • 0x1= HPS only boots from the FPGA; BSEL options are ignored and CSEL fuse options are ignored.
5	aes_en_f	This fuse value indicates if a decryption of the flash image is always performed. • 0x0= An AES decryption of the flash image is determined from the second stage boot loader header. • 0x1= An AES decryption of the flash image is always performed.
4:2	kak_src_f	This bit field indicates the source of the Key Authorization Key (KAK) which can be in: • Proprietary ROM • FPGA logic elements • User fuses
1	kak_len_f	This fuse value indicates the Key Authorization Key (KAK) length: • 0x0= 256 bits • 0x1= 384 bits
0	authen_en_f	This fuse value determines whether authentication of flash images is required prior to execution. • 0x0= No authentication of the flash image is required prior to execution. • 0x1= HPS authentication of all flash images is required prior to execution.

At initialization, the FPGA also receives fuse information that is pertinent to its configuration. The HPS can read this information through a secure serial interface, which shifts the FPGA fuse values into the

fpga_fusesec register in the Security Manager. The CSS shifts in a 32-bit value although some of the bits are considered reserved.

Related Information

- **Clock Configuration** on page 6-12
  For more information about the CSEL settings, refer to the "Clock Configuration" section.
- **Configuration, Design Security, and Remote System Upgrades in Arria 10 Devices**

Security State

The security state of the entire device is defined by the combination of the security state of the HPS and the FPGA.The initial security state of the device is determined by the fuse settings in the SoC. The values of these fuses are written to two shadow registers within the Security Manager: the hps_fusesec and the fpga_fusesec registers.

Each feature's security level can be increased from its initial fuse level through software, and for debug, through hardware as well. Changes to security level can only increase security and can never lower security. Both the HPS and FPGA have a mechanism to load security option bits as part of the second-stage boot loader and POF file, respectively, which can perform a check of the current security state and raise the security level.

Security Check

The Security Manager has the capability to perform security checks to identify discrepancies between the actual and expected security level so that measured actions can be taken through software and hardware if required.

There are three types of security checks that can be performed by the Security Manager:

- A fuse security level check
- A hardware access security level check
- A software access security level check

Secure Serial Interface

The HPS and FPGA communicate through a secure serial interface. The interface allows security of the HPS to be separate from the security of the FPGA. Software can initiate a request of the serial fuse interface via the Security Manager register.

Secure Debug

Debug logic in the Security Manager controls whether to enable or disable the JTAG, CPUs, Debug Access Port (DAP) and CoreSight™ domains. Debug access is determined by a combination of debug fuses, option bits, and registers that change the default state and set the functional state of debug peripherals.

During a power-on reset, access to all debug domains are prevented. During a warm reset, access to all debug domains except JTAG are prevented. Once a cold or warm reset is exited, the default debug access is defined by the value of the dbg_disable_access fuse and the dbg_lock* fuses. When the dbg_disable_access fuse is blown, HPS debug accesses to all domains are disabled by default out of reset. If the fuse is not blown, then the initial HPS debug state out of reset is enabled. To ensure that the device is brought up in a secure state, the dbg_disable_access fuse must be blown.

Related Information

- **ARM Debug Permissions**
  Refer to this location for information on debug signals and permissions.
- **ARM Infocenter**
  For more information about debug, refer to the ARM Cortex-A9 Technical Reference Manual.

MPU

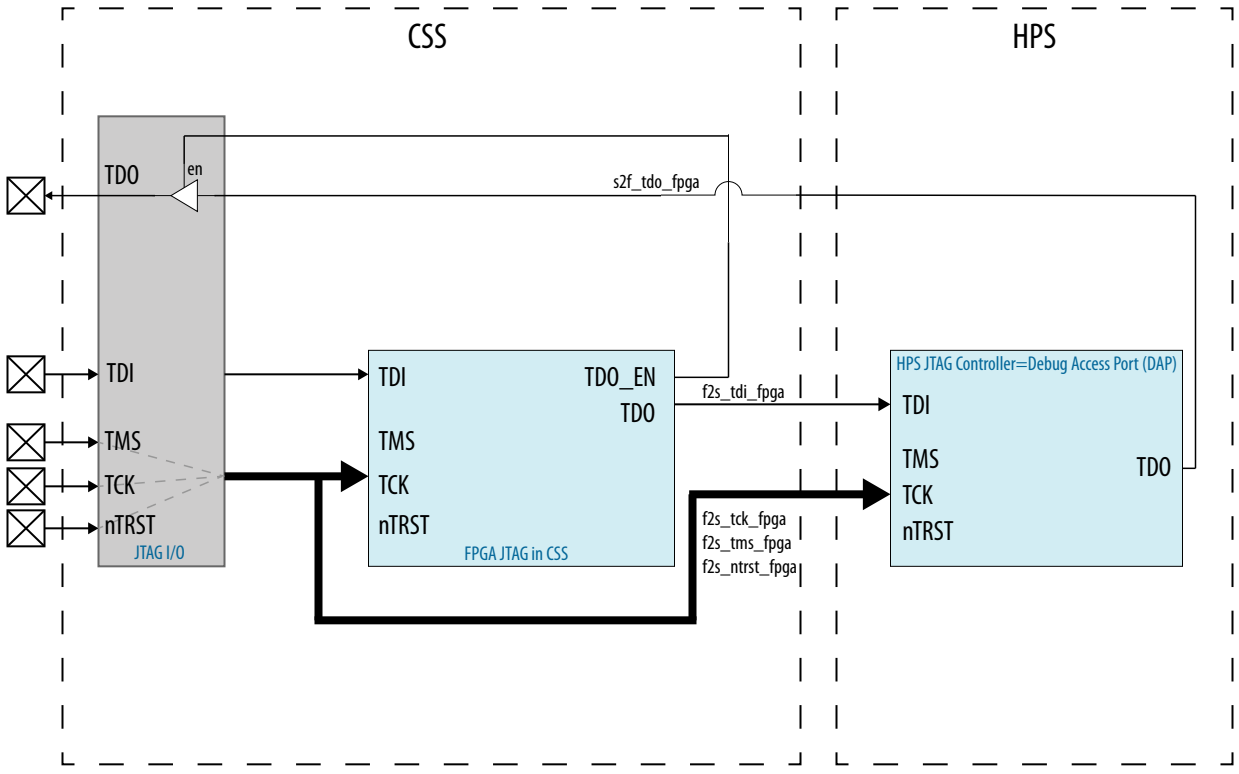
The MPU supports invasive and non-invasive debug. Invasive debug allows the user to control and observe the processor. Non-invasive debug allows you to observe the processor but not control it and is always available as part of invasive debug. The `NIDEN` and `SPNIDEN` signals are used for non-invasive debug in the MPU.

JTAG

The HPS and FPGA share a common set of JTAG pins and each have their own TAP controller which are chained together inside the device.

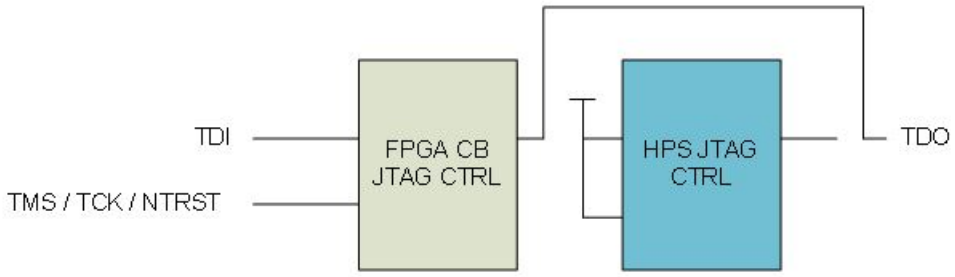
In the default JTAG configuration, the FPGA and HPS JTAG controllers are daisy-chained and controlled by an external JTAG port. The HPS JTAG controller is also referred to as the Debug Access Port.

Figure 6-3: JTAG Chaining



During power-on-reset, the JTAG and all debug domains are disabled. After the device is released from reset, the debug fuses are read by the Configuration Subsystem to determine if the external JTAG port or the JTAG controller to the FPGA or HPS is bypassed. Bypass can additionally be enabled through option bits in the Security Option registers discussed in the *Security State* section.

Figure 6-4: HPS JTAG Bypass



Reset Manager

To avoid unsecure snooping of RAM contents, the Security Manager can send signals to the Reset Manager to clear all RAM on a cold or warm reset.

These signals are the result of fuse programming combined with the configuration bits found in the `HPS_Swoptset` and `HPS_secopt1` register in the Security Manager.

In addition, the fuse information that determines whether the RAMs are cleared in series or parallel is also sent to the Reset Manager. Clearing the memory in series prevents a power spike that could occur with parallel clearing of RAM. The policies for how or when security states take affect can be programmed in the `HPS_secopt1` register. Refer to the "Security State" section for more information.

The RAM instances that are cleared during a warm or cold reset are:

- On-chip RAM
- USB0
- USB1
- SD/MMC
- EMAC0 RX and TX buffer
- EMAC1 RX and TX buffer
- EMAC2 RX and TX buffer
- DMA
- NAND read data, write data, and ECC RAMs
- QSPI
- MPU RAM

If a tamper event occurs, the Security Manager notifies the Reset Manager and the following sequence occurs:

1. All domain resets are asserted.
2. All RAMs are cleared.
3. The Reset Manager enters a dead state waiting for a POR release from Security Manager. All cold reset sources except Security Manager are ignored.

When FPGA POR and HPS POR are asserted, the Security Manager goes into reset.

Related Information

- **Security State** on page 6-9
  For more information regarding Security States, refer to the "Security State" section.
- **Reset Manager** on page 3-1
  For more information about reset, refer to the *Reset Manager* chapter.

Clock Configuration

Security Manager is driven by an internal oscillator, `cb_intosc_hs_clk`, in the Configuration Subsystem. This clock has wide variation across process and temperature. Once the Security Manager reads the fuse information, it drives the boot clock configuration to the Clock Manager. The Clock Manager samples the clock configuration on a cold and warm reset. If the `hps_clk_f` fuse is blown, the internal oscillator divided by 2, `cb_intosc_hs_div2_clk`, is used as the boot clock. If the fuse is not blown, an external oscillator is used. During boot ROM execution, the boot code configures the device clock based on the CSEL fuse settings or software code. The `cb_intosc_hs_div2_clk` can be considered the secure reference clock option.

The CSEL fuse settings have different meanings depending on how the hps_clk_f fuse is configured. When the hps_clk_f fuse is not blown, the external oscillator is the boot clock input.

Table 6-2: CSEL Encodings for hps_clk_f = 0

CSEL[3:0] Fuse Value	Description	MPU Clock Value
0x0	When no fuses are blown, the boot ROM returns clocks back to their default (bypass) boot mode and the CPU is driven by the external oscillator (`EOSC1`), which must be in the range of 10 to 50 MHz. No PLL is enabled.	External Oscillator, `EOSC1` (10 to 50 MHz)
0x1	This encoding is typically used during a warm reset if you have already configured your clocks or PLL in the clock manager and would like to continue to use this configuration. If this is encoding is selected, no changes are made to these settings and the clock configuration is essentially user selected. If this is encoding is used during a power-on reset, then whatever the default reset values of the clocks are used.	User-selected clock source. Refer to the *Clock Manager* chapter for clock register settings.

CSEL[3:0] Fuse Value	Description	MPU Clock Value
0x2	Reserved	External Oscillator, EOSC1 (10 to 50 MHz)
0x3	Reserved	External Oscillator, EOSC1 (10 to 50 MHz)
0x4	Reserved	External Oscillator, EOSC1 (10 to 50 MHz)
0x5	Reserved	External Oscillator, EOSC1 (10 to 50 MHz)
0x6	Reserved	External Oscillator, EOSC1 (10 to 50 MHz)
0x7	External oscillator input clock, EOSC1, is in the range of 10 to 15 MHz	533 to 800Mhz
0x8	External oscillator input clock, EOSC1, is in the range 15 to 20 MHz.	600 to 800Mhz
0x9	External oscillator input clock, EOSC1, is in the range 20 to 25 MHz.	640 to 800Mhz
0xA	External oscillator input clock, EOSC1, is in the range 25 to 30 MHz.	666 to 800Mhz
0xB	External oscillator input clock, EOSC1, is in the range 30 to 35 MHz.	685 to 800Mhz
0xC	External oscillator input clock, EOSC1, is in the range 35 to 40 MHz.	700 to 800Mhz
0xD	External oscillator input clock, EOSC1, is in the range 40 to 45 MHz.	711 to 800Mhz
0xE	External oscillator input clock, EOSC1, is in the range 45 to 50 MHz.	720 to 800Mhz
0xF	The boot ROM returns clocks back to their default (bypass) boot mode and the CPU is driven by the external oscillator (OSC1). No PLL is enabled.	External Oscillator (10 to 50 MHz)

Table 6-3: CSEL Encodings for hps_clk_f = 1

CSEL[3:0] Fuse Value	Description	MPU Clock Value
0x0	Bypass mode; in this mode all clocks are reset and boot mode is ensured active; `cb_intosc_hs_div2_clk` (`cb_intosc_hs` clock divided by 2) is used.	`cb_intosc_hs_div2_clk` (60 to 200 MHz)
0x1	This encoding is typically used during a warm reset if you have already configured your clocks or PLL in the Clock Manager and would like to continue to use this configuration. If this is encoding is selected, no changes are made to these settings and the clock configuration is essentially user selected. If this is encoding is used during a power-on reset, then whatever the default reset values of the clocks are used.	User-selected clock source. Refer to the *Clock Manager* chapter for clock register settings.
0x2	PLL is used with the internal oscillator (30 to 100 MHz)	120 to 800 MHz
0x3	Reserved	`cb_intosc_hs_div2_clk` (60 to 200 MHz)
0x4	Reserved	`cb_intosc_hs_div2_clk` (60 to 200 MHz)
0x5	Reserved	`cb_intosc_hs_div2_clk` (60 to 200 MHz)
0x6	Reserved	`cb_intosc_hs_div2_clk` (60 to 200 MHz)
0x7	Reserved	`cb_intosc_hs_div2_clk` (60 to 200 MHz)
0x8	Reserved	`cb_intosc_hs_div2_clk` (60 to 200 MHz)
0x9	Reserved	`cb_intosc_hs_div2_clk` (60 to 200 MHz)
0xA	Reserved	`cb_intosc_hs_div2_clk` (60 to 200 MHz)
0xB	Reserved	`cb_intosc_hs_div2_clk` (60 to 200 MHz)
0xC	Reserved	`cb_intosc_hs_div2_clk` (60 to 200 MHz)
0xD	Reserved	`cb_intosc_hs_div2_clk` (60 to 200 MHz)
0xE	Reserved	`cb_intosc_hs_div2_clk` (60 to 200 MHz)

CSEL[3:0] Fuse Value	Description	MPU Clock Value
0xF	Bypass mode; reset all clocks and ensure boot mode is active; `cb_intosc_hs_div2_clk` is used.	`cb_intosc_hs_div2_clk` (60 to 200 MHz)

For more information regarding the CSEL encodings and descriptions, please refer to the *Booting and Configuration* Appendix.

Related Information

- **Clock Manager** on page 2-1
  For more information about the Clock Manager.
- **Booting and Configuration** on page 30-1
  For more information about the boot process refer to the *Booting and Configuration* appendix.
- **Arria 10 Device Datasheet**
  For more information regarding clock characteristics, refer to the *Arria 10 Device Datasheet*.

FPGA Security Features

The FPGA has several security fuses which can control the following functions:

- Limiting acceptance of only encrypted and authenticated POFs
- Disabling partial reconfiguration and scrubbing
- Disabling test access
- Disabling external configuration
- Limiting JTAG access
- Bypassing HPS or FPGA JTAG
- Locking or disabling the battery-backed volatile key
- Disabling the OTP key

Note that there is a fuse sent to the FPGA CSS, called hps_jtag_bypass, which performs the same function of removing the HPS from the JTAG chain. However, this fuse is a permanent disable, whereas using software to program the sec_jtagdbg register has the same affect, but may be changed later by software.

Secure Boot

No ordering of boot or FPGA configuration is required because the FPGA and HPS may be brought up in any order, and they can both raise the security level of the entire device at any time before user code is loaded to execute.

The SoC may securely boot in one of three ways:

1. The HPS boot and FPGA configuration occur separately.
2. The HPS boots from the FPGA after the FPGA is configured
3. The HPS boots first and configures the FPGA.

Note: The device has the capability to execute standard non-secure boot in any of these three ways, as well. In this section, only secure booting mechanisms are reviewed.

Note: The FPGA must be powered on for the HPS to come out of reset properly.

Related Information

Booting and Configuration on page 30-1
For more information about the boot process refer to the *Booting and Configuration* appendix.

Boot Configuration

Booting configuration is provided from a combination of the `BSEL` pins and the boot mode fuses. The `BSEL` pins indicate if the boot source is the FPGA or one of the HPS flash interfaces. If the `fpga_boot_f` fuse is blown, it overrides these pin values. The `fpga_boot_f` fuse value can be read from the `hps_fusesec` register.

Table 6-4: BOOTSEL Field Values and Flash Device Selection

BOOTSEL Field Value	Flash Device
0x0	Reserved
0x1	FPGA (HPS-to-FPGA bridge)
0x2	1.8 V NAND flash memory
0x3	3.0 V NAND flash memory
0x4	1.8 V SD/MMC flash memory with external transceiver
0x5	3.0 V SD/MMC flash memory with internal transceiver
0x6	1.8 V quad SPI flash memory
0x7	3.0 V quad SPI flash memory

Within the device, there are fuses that provide some of the boot mode configuration information. The fuse values can be read through the `HPS_fusesec` register:

Table 6-5: `HPS_fusesec` Register Description

Bits	Name	Description
31:27	`Reserved`	Bit values in this field are undefined.
26:23	`csel_f`	This field indicates the value of the clock select fuses that are available for configuring the clock for the boot interface and for the PLLs. Refer to the *Clock Configuration* section for more information on CSEL encodings.
22	`dbg_access_f`	This fuse determines the initial state of the debug access domains.

Bits	Name	Description
21	dbg_lock_JTAG	This field indicates if the HPS JTAG access level can be changed through software when the HPS is released from reset. • 0x0= HPS JTAG access level can be changed through the sec_jtagdbg register. • 0x1= HPS JTAG access level cannot be changed (locked).
20	dbg_lock_DAP	This field indicates if the DAP access level can be changed through software when the HPS is released from reset. • 0x0= The DAP access level can be changed through the sec_dapdbg register. • 0x1= The DAP access level cannot be changed (locked).
19	dbg_lock_CPU0	This field indicates if the CPU0 debug access level can be changed through software when the HPS is released from reset. • 0x0= CPU0 debug access level can be changed through the sec_cpu0dbg register. • 0x1= CPU0 debug access level cannot be changed (locked).
18	dbg_lock_CPU1	This field indicates if the CPU1 debug access level can be changed through software when the HPS is released from reset. • 0x0= The CPU1 debug access level can be changed through the sec_cpu1dbg register. • 0x1= The CPU1 debug access level cannot be changed (locked).
17	dbg_lock_CS	This field indicates if the CoreSight debug access level can be changed through software when the HPS is released from reset. • 0x0= The CoreSight debug access level can be changed through the sec_csdbg register. • 0x1= The CoreSight debug access level cannot be changed (locked).

Bits	Name	Description
16	dbg_lock_FPGA	This field indicates if the FPGA debug access level can be changed through software when the HPS is released from reset. • 0x0= The FPGA debug access level can be changed through the sec_fpgadbg register. • 0x1= The FPGA debug access level cannot be changed (locked).
15:12	Reserved	Bit values in this field are undefined.
11	clr_ram_order_f	This fuse value determines how RAMs are cleared during a tamper event. • 0x0= All RAMs are cleared in parallel. • 0x1= All RAMs are cleared in series.
10	clr_ram_cold_f	This fuse value indicates what happens to the RAM on a cold reset. • 0x0= All RAMs are not cleared on a cold reset. • 0x1= All RAMs are cleared on a cold reset.
9	clr_ram_warm_f	This fuse value indicates what happens to the RAMs on a warm reset. • 0x0= All RAMs are not cleared on a warm reset. • 0x1= All RAMs are cleared on a warm reset.
8	oc_boot_f	This fuse value determines if the second-stage boot code is allowed to boot from on-chip RAM. • 0x0= Second-stage boot can be from on-chip RAM if enabled by the System Manager. • 0x1= Second-stage boot is not from on-chip RAM.
7	hps_clk_f	This fuse value selects the clock used for the boot process and in the case of a tamper event, memory scrambling. • 0x0= The external oscillator, EOSC1, is used for boot. • 0x1= The internal oscillator, cb_intosc_ls_clk, is used for boot.

Bits	Name	Description
6	fpga_boot_f	If blown, this fuse value allows the FPGA to configure independently and allows the HPS to boot from an encrypted next-stage boot source that was decrypted into the FPGA. • 0x0= Booting is dependent on the BSEL pins. • 0x1= HPS only boots from the FPGA; BSEL options are ignored and CSEL fuse options are ignored.
5	aes_en_f	This fuse value indicates if a decryption of the flash image is always performed. • 0x0= An AES decryption of the flash image is determined from the second stage boot loader header. • 0x1= An AES decryption of the flash image is always performed.
4:2	kak_src_f	This bit field indicates the source of the Key Authorization Key (KAK) which can be in: • Proprietary ROM • FPGA logic elements • User fuses
1	kak_len_f	This fuse value indicates the Key Authorization Key (KAK) length: • 0x0= 256 bits • 0x1= 384 bits
0	authen_en_f	This fuse value determines whether authentication of flash images is required prior to execution. • 0x0= No authentication of the flash image is required prior to execution. • 0x1= HPS authentication of all flash images is required prior to execution.

Secure Boot Process

Secure boot allows the HPS to release from reset into a known state and then validate the authenticity and integrity of the second-stage boot loader code prior to executing it. Secure boot ensures that only authorized code is executed on the system. In addition, the system has the option to configure the FPGA from the HPS, which provides a secure boot mechanism for the FPGA portion of the SoC. In this mode, the HPS boot code can authenticate the FPGA POF prior to loading it.

The HPS and FPGA can also be booted securely but independently, with the HPS executing authenticated and decrypted user code out of external RAM and the FPGA receiving an encrypted FOP configuration file by enabling an FPGA configuration source through the MSEL pins.

Boot ROM code runs on CPU0 and CPU1 is held in reset. Once control passes to the second-stage boot loader, CPU1 can be released from reset.

Authentication and Decryption

Authentication is provided for the second-stage boot loader code and both the HPS and FPGA can utilize the AES algorithms in the Configuration Subsystem (CSS) to decrypt boot images and POF files, respectively.

Three levels of boot are available to the device:

- Authentication only: The second-stage boot loader code is not encrypted, but there are public key signatures attached to the image and the code only executes if all of the signatures pass. ECDSA256 (SHA 256) is used for authenticated boot.
- Decryption only: The user boot code is encrypted and must be decrypted before execution. AES-based algorithms in the FPGA are used for decryption.
- Authentication and Decryption: The user boot code is encrypted and signed.

If authentication and decryption are enabled, the data is first authenticated and then decrypted using the AES algorithms. Authentication is performed using the public key authorization key (KAK) held in the user fuses. The KAK can be 256 bits. The KAK public key authentication fuses are lockable by the user in groups of 64 bits or less.

The Security Manager's hps_fusesec register can be read to identify the source of the public (root) key used for authentication of the second-stage boot loader. The source of the public (root) key can be one of the following:

- External software
- Altera ROM
- FPGA logic elements
- User Access Fuses (UAF)

AES decryption algorithms are available through the FPGA CSS and are enabled prior to decryption through a combination of user fuses and software programming. Elliptic curve cryptographic algorithms are used in an authenticated boot. Decrypted data is sent to the Security Manager to be read by the CPU or DMA.

The FPGA has higher priority in the CSS, and the FPGA AES algorithms abort decryption and authentication of data if the FPGA starts configuration or reconfiguration.

Anti-Tamper

The Security Manager receives a tamper detection signal when a tamper event is detected in the device.

Anti-tamper logic can be implemented in the FPGA soft logic to detect a tamper event. No tamper detection is available until the FPGA is configured. The user must define the anti-tamper design. Some examples of tamper detection that could be implemented are:

- Using the FPGA voltage sensor to detect any IR drops across the device.
- Using an A/D controller to detect temperature changes in the device due to tamper.
- Assigning loaner I/O as dedicated tamper pins to detect package break-away.

The tamper detection signal sent to the Security Manager, `anti_tamper_in`, is an active low signal. When there is no tamper event, it is driven to 1. The Security Manager also provides a tamper acknowledge signal to the FPGA interface. The tamper event input to the HPS is gated with a signal indicating the completion of FPGA configuration. This prevents an inadvertent tamper event assertion or tamper response from the HPS to the rest of the system.

When the Security Manager receives a tamper detection assertion, it prepares the system to go into reset and notifies the Reset Manager of the event. When the Reset Manager receives the tamper event assertion from the Security Manager, it drives all the peripherals and MPU into reset. It asserts a request and enables all peripherals memories to initiate a memory scramble and a memory clear, either in series or parallel, depending on the information sent from the Security Manager. The memory clearing includes the on-chip RAM, peripheral memories, L1 caches, and L2 cache. Once the memory scramble and clear completes, each module sends an acknowledgment back to the Reset Manager.

Upon completion of memory scrambling of all IPs, the Security Manager sends a response to the FPGA indicating that a tamper event has been handled in the HPS. This response is followed by HPS entering into a cold reset without exiting until another power cycle occurs. Both Security Manager and Reset Manager enter reset. Other resets, such as warm reset or cold reset, do not affect the state of the Security Manager or the Reset Manager.

If the FPGA POR is de-asserted while the HPS is in a tamper event reset, then the Security Manager goes back to the reset state.

Security System Extensions

Beyond the Security Manager module, other secure features can be implemented at a system level within the SoC.

TrustZone

The Cortex-A9 MPU subsystem has integrated TrustZone technology, which provides a system solution to protect application platforms from malicious attack. The TrustZone hardware and supporting software are designed to provide a strong security solution regardless of the operating environment. TrustZone creates a separation between the secure and non-secure areas of the SoC and allows the designer to choose which assets in a design are designated as secure and non-secure.

TrustZone security is implemented in the Cortex-A9 MPU subsystem in three ways:

- Hardware partitioning through the implementation of firewalls: Resources can be assigned and identified as secure or non-secure.
- Virtual processor execution: Each core can context switch between secure and non-secure operation.
- Secure debug control: Both secure and non-secure hardware debug exist and the type of debug allowed can be configured by the user.

Secure Partitioning

System interconnect and cache accesses as well as processor execution can be securely partitioned using the TrustZone infrastructure.

Secure Bus Accesses

The Cortex-A9 MPCore and other masters that utilize the system interconnect can be configured for secure or non-secure accesses.

Master accesses drive a protection signal, `AxPROT[1]`, to communicate the security level of the attempted transaction. The Cortex-A9 drives this signal low for secure accesses and high for non-secure accesses. The secure firewalls determine whether the access is allowed or not. A slave access type defaults to secure if no other configuration is programmed. Users can define whether a bus error is generated or if a bus failure occurs when a secure access is violated.

Note: Secure processor configuration is programmed through the `CP15` register.

Related Information

- **Arria 10 HPS Secure Firewalls** on page 6-24
  For more detailed information about secure bus accesses, refer to the "Secure Firewall" section.
- **SoC Security** on page 6-1
  For more information about SoC Security, refer to the *SoC Security* Chapter.

Secure Cache Accesses

Secure and non-secure partitioning also extends to the L1 and L2 caches.

There is a 32-bit physical address space for secure and non-secure transactions and a 33rd bit is provided to indicate security. The cache is able to store both secure and non-secure lines.

The Accelerator Coherency Port (ACP) in the ARM Cortex-A9 MPCore can be used with TrustZone technology. The security state of the masters using the ACP must be the same as the Cortex-A9 processor.

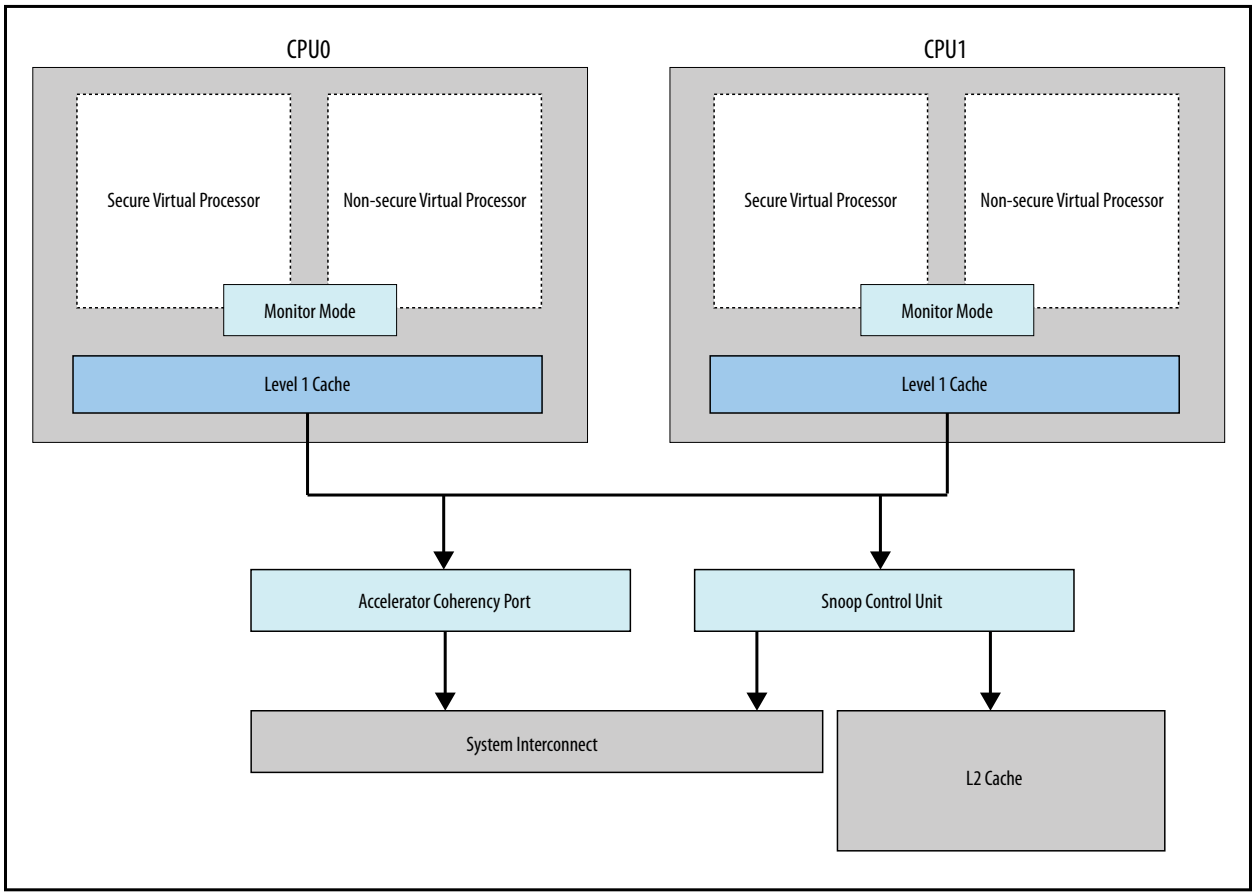
Related Information

- **ARM TrustZone**
  Refer to this location for information about virtual core operation.
- **SoC Security** on page 6-1
  For more information about SoC Security, refer to the *SoC Security* Chapter.

Virtual Processor Operation

Two virtual processors (secure and non-secure) with context switch capability (monitor mode) exist for each Cortex-A9 processor core. The secure virtual processor only accesses secure resources and the non-secure virtual processor only accesses non-secure resources.

Figure 6-5: Virtual Processor Environment with Monitor Mode



A context switch to secure operation is achieved through a secure monitor call (SMC) instruction or the following hardware exceptions (if configured to do so):

- IRQ interrupt
- Faster Interrupt Request (FIQ) interrupt
- External data abort
- External prefetch abort

When a context switch occurs, the state of the current mode is saved and restored on the next context switch or on a return from exception.

Exception Vector Tables

Three exception vector tables are provided for the MPU with TrustZone:

1. Non-secure
2. Secure
3. Monitor mode

Typically IRQs are used for non-secure interrupts and FIQs are used for secure interrupts. The location of each of the three vector tables can be moved at runtime.

The Generic Interrupt Controller (GIC) can handle secure and non-secure interrupts and prevent non-secure accesses from reading or modifying the configuration of a secure interrupt. An interrupt can be made secure by programming the appropriate bits in the Interrupt Security Register. Secure interrupts are always given a higher priority than non-secure interrupts.

Arria 10 HPS Secure Firewalls

You can use the system interconnect firewalls to enforce security policies for slave and memory regions in the system interconnect.

The firewalls support the following features:

- Secure or non-secure access configurable per peripheral
- Privileged or user access configurable for some peripherals
- Per-transaction security

Each firewall contains the security configuration registers (SCRs) that set security policies and define which transactions are allowed to go through the firewall. If you want to generate an error any time a transaction is blocked in the firewall, then you must set the `error_response` bit in the `global` register of the `noc_fw_ddr_l3_ddr_scr` module at base address 0xFFD13400. If this bit is clear, transactions blocked by the firewall return random data.

Note: Future devices may not support the return of random data and may only support an error response for blocked firewall transactions. For designs that may be ported to future devices, Altera recommends you to set the `error_response` bit in the `global` register.

There are five main firewalls in the HPS:

- Peripheral
- System
- HPS-to-FPGA
- On-Chip RAM
- SDRAM (which includes DDR and DDR L3 firewalls)

Related Information

- **System Interconnect** on page 7-1
  For more information about the System Interconnect, refer to the *Interconnect* chapter.
- **SoC Security** on page 6-1

Firewall Diagrams

The system interconnect firewalls filter access to components on various buses.

Table 6-6: System Interconnect Firewalls

Name	Function
Peripherals	The peripherals firewall filters access to slave peripherals in the following buses: • L4 main bus • L4 master peripheral bus • L4 AHB bus • L4 slave peripherals bus
System	The system firewall filters access to system peripherals in the following components: • L4 system bus • L4 ECC bus • DAP
HPS-to-FPGA	The HPS-to-FPGA firewall filters access to FPGA through the following bridges: • HPS-to-FPGA bridge • Lightweight HPS-to-FPGA bridge
On-Chip RAM	The on-chip RAM firewall filters secure access to on-chip RAM
SDRAM	The SDRAM firewall filters access to DDR SDRAM

Figure 6-6: Peripheral Firewall

Security Configuration Registers (SCRs) within the `noc_fw_l4_per` register group can be programmed to mark the security status of all available masters.

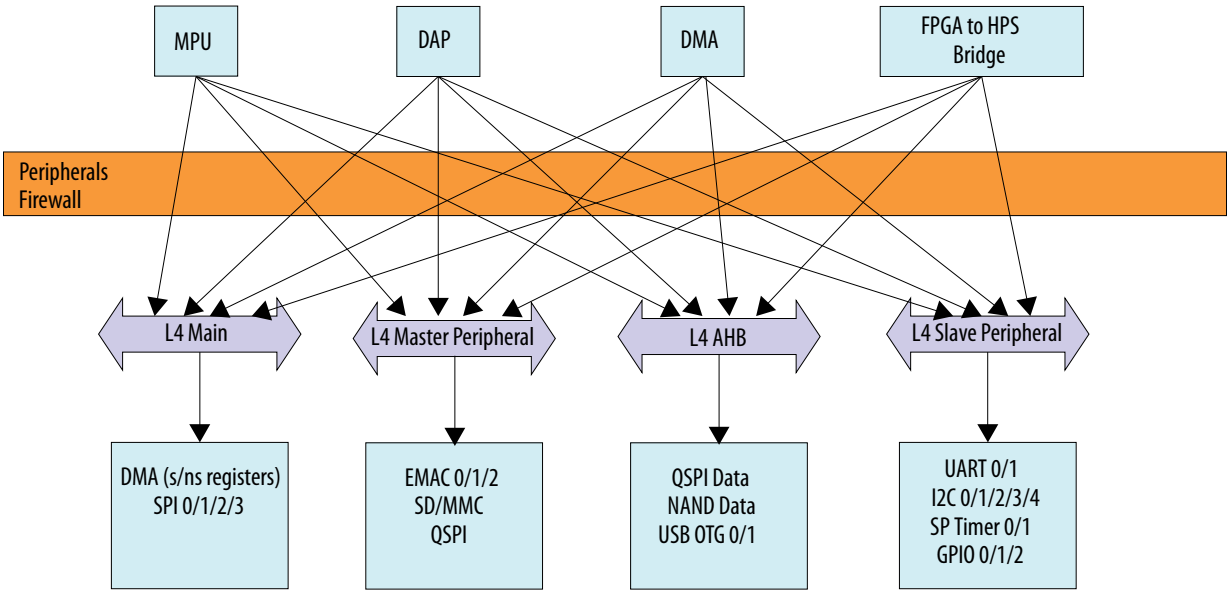


Table 6-7: Peripheral Firewall

Bus Behind Firewall	Bus Slaves Behind Firewall	Masters That Can Access Bus
L4 Main	DMA (secure/non-secure registers) SPI 0/1/2/3	MPU DAP DMA FPGA-to-HPS
L4 Master Peripheral Bus	EMAC 0/1/2 SD/MMC QSPI	MPU DAP DMA FPGA-to-HPS
L4 AHB Bus	QSPI Data NAND Data USB OTG 0/1	MPU DAP DMA FPGA-to-HPS

Bus Behind Firewall	Bus Slaves Behind Firewall	Masters That Can Access Bus
L4 Slave Peripheral Bus	UART 0/1	MPU
	I2C 0/1/2/3/4	DAP
	SP Timer 0/1	DMA
	GPIO 0/1/2	FPGA-to-HPS

Figure 6-7: System Firewall

The system firewall filters accesses to all system peripherals. The system firewall policies can be programmed through the SCRs of the `noc_fw_soc2fpga` register group.

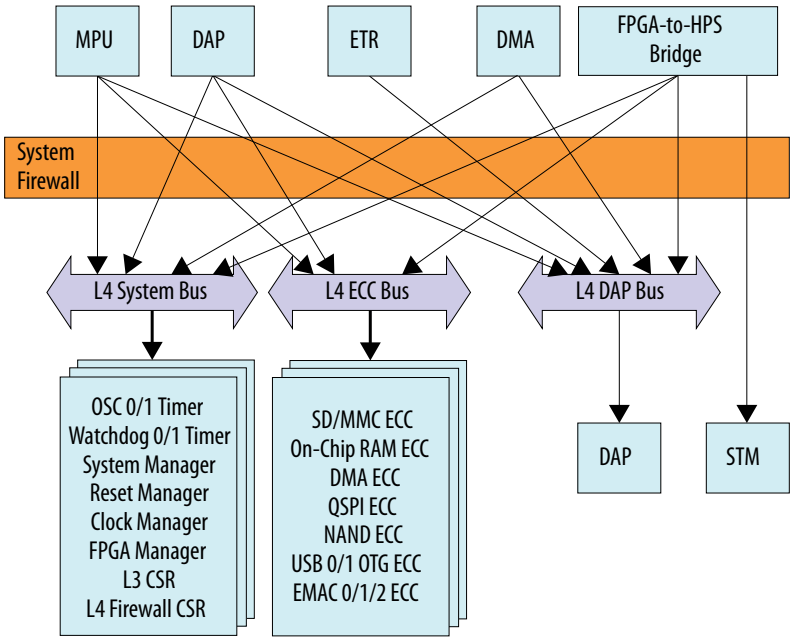


Table 6-8: System Firewall

Bus Behind Firewall	Bus Slaves Behind Firewall	Masters That Can Access Bus
L4 ECC	SD/MMC ECC DMA ECC QSPI ECC NAND ECC USB 0/1 ECC On-Chip RAM ECC EMAC 0/1/2 Rx ECC EMAC 0/1/2 Tx ECC NAND Read/Write ECC	MPU FPGA-to-HPS DAP
L4 System	FPGA Manager Data and Registers OSC Timer 0/1 Watchdog 0/1 System Manager L3 Interconnect Control and Status Registers (CSR) L4 Interconnect firewall Security Control Registers (SCR)	MPU FPGA-to-HPS DAP DMA
L4 DAP Bus	STM DAP	MPU FPGA-to-HPS DAP

Figure 6-8: HPS-to-FPGA Firewall

SCRs within the `noc_fw_soc2fpga` register group can be programmed to configure the security policy for the master-to-bridge pairs.

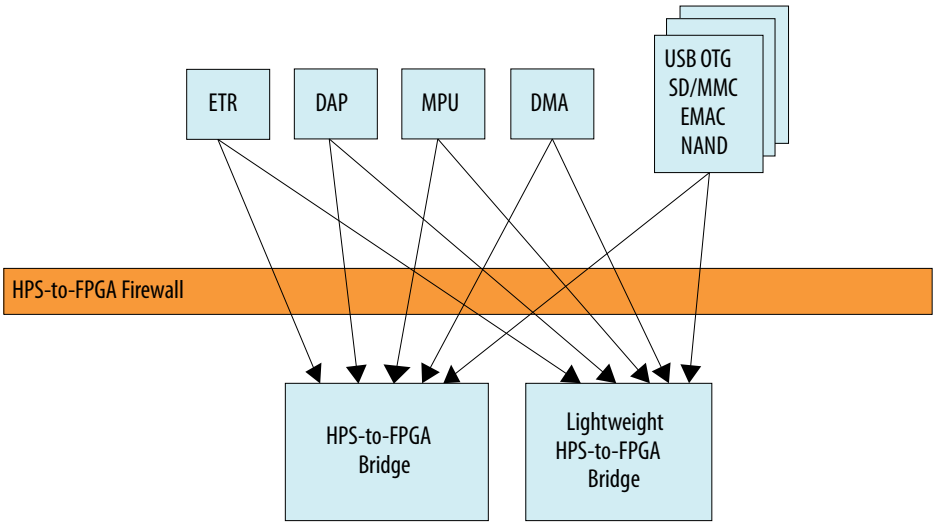


Table 6-9: System Firewall

Bus Behind Firewall	Bus Slaves	Masters that can Access Bus
L3	SD/MMC ECC DMA ECC QSPI ECC NAND ECC USB 0/1 ECC On-Chip RAM ECC EMAC 0/1/2 Rx ECC EMAC 0/1/2 Tx ECC NAND Read/Write ECC	MPU FPGA-to-HPS DAP
L4 System	FPGA Manager Data and Registers OSC Timer 0/1 Watchdog 0/1 System Manager L3 Interconnect CSR L4 Interconnect firewall SCR	MPU FPGA-to-HPS DAP DMA

Bus Behind Firewall	Bus Slaves	Masters that can Access Bus
L4 DAP Bus	STM DAP	MPU FPGA-to-HPS DAP

Figure 6-9: On-Chip RAM Firewall

Up to six regions of the on-chip RAM can be partitioned for non-secure accesses.

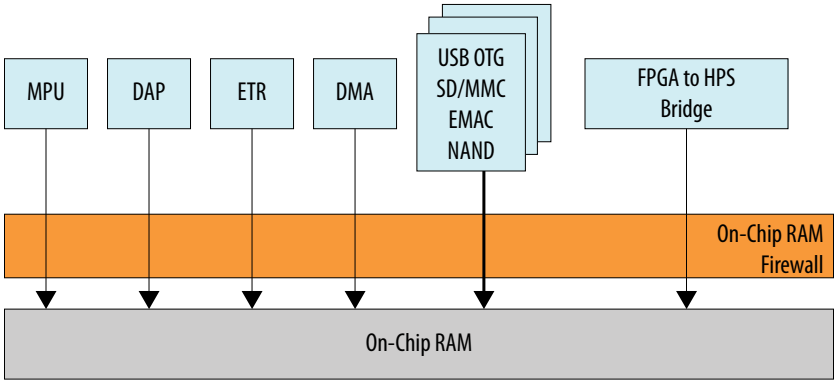
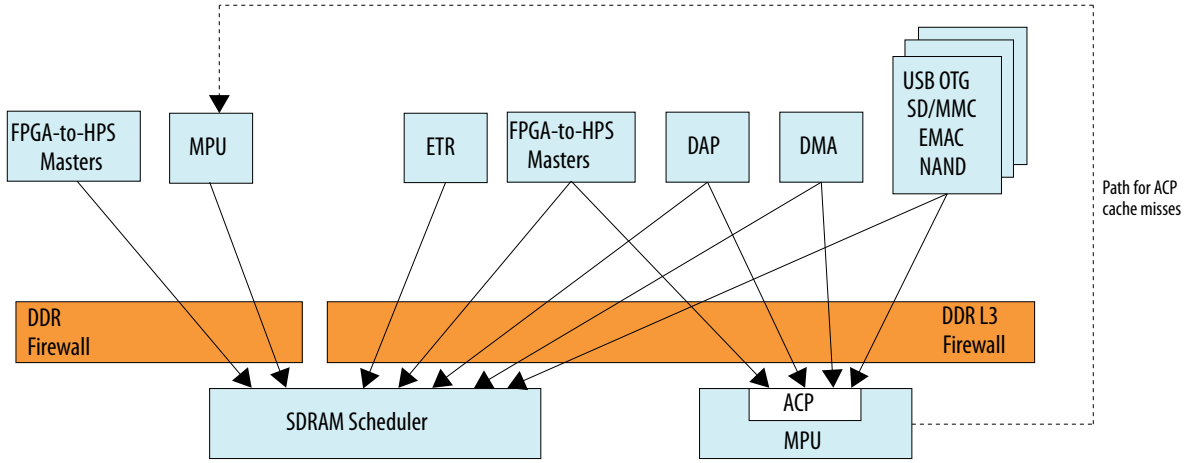


Figure 6-10: SDRAM firewall

Transactions from the DMA, ETR, DAP, FPGA-to-HPS masters or HPS masters (USB, SD/MMC, EMAC, NAND) are routed to either the ACP or the SDRAM scheduler depending on the cacheable bit. Only cacheable transactions are routed to the ACP, with non-cacheable transactions being routed directly to the SDRAM scheduler. If an access to the ACP results in a cache miss in the L1 and L2 cache systems, then the MPU master M1 issues a transaction to the SDRAM scheduler. Only cacheable transactions go through to the ACP. If a cache miss occurs on the ACP cycle, then the MPU masters a new transaction to the SDRAM scheduler. Accesses from the MPU or FPGA-to-SOC masters that are tightly coupled to the MPU go through the DDR firewall only.

Note: The dotted line in the diagram represents the path taken when an ACP access occurs and misses in the L1 and L2 cache.



Secure Transaction Protection

The interconnect provides two levels of transaction protection.

The two levels of transaction protection are:

- Security Firewalls:

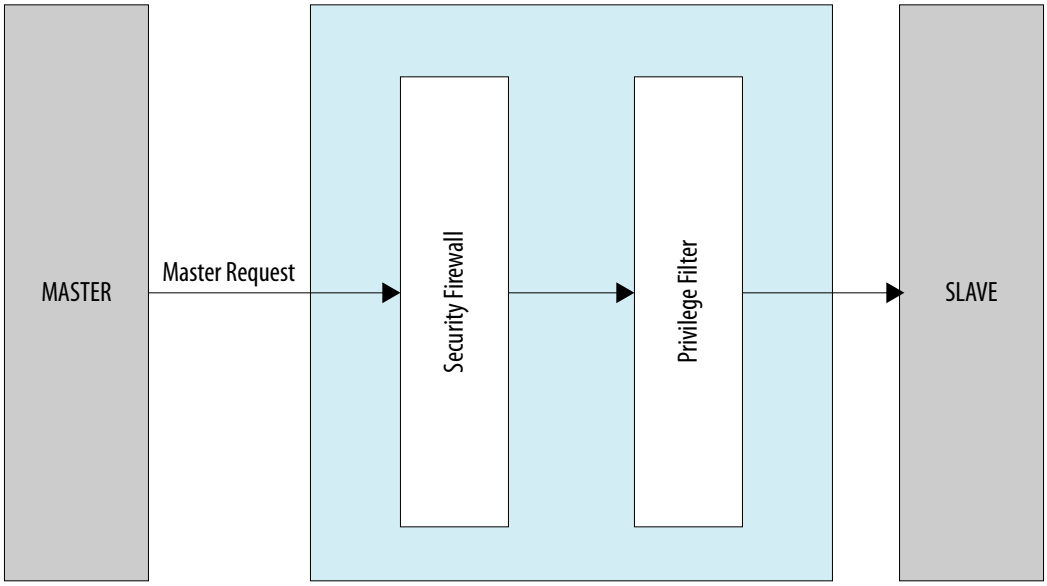
  Security firewalls enforce the Secure World read and write transactions. The term "security firewall" and "firewall" may be used interchangeably within the context of this section.
- Privilege filter

  The privilege filter leverages the firewall mechanism and provides additional filtering by allowing control of the privilege level of L4 slave transactions. The privilege filter applies to writes only.

All slaves on the SoC are placed behind a security firewall. A subset of slaves are also placed behind a privilege filter. Transactions to these slaves pass through two firewall structures as shown in the figure below. The transaction packet must pass both a security firewall and the privilege filter to reach the slave. A transaction error causes random data or a slave error depending on where the transaction failed and how the `error_response` bit in the `global` register of the `noc_fw_ddr_l3_ddr_scr` module is programmed.

Note: Future devices may not support the return of random data and may only support an error response for blocked firewall transactions. For designs that may be ported to future devices, Altera recommends you to set the `error_response` bit in the `global` register.

Figure 6-11: Successful Secure Transaction Path



Each firewall has an associated firewall register which can be programmed by the DAP, FPGA fabric or MPU in secure mode only. A subset of the L4 slaves can additionally be programmed to configure their privilege filter policy.

Peripheral and System Firewalls

The peripheral and system firewalls each have a group of Security Configuration Registers (SCRs) that can be programmed by the DAP, FPGA fabric or MPU only in secure mode.

The peripheral and system firewalls can be configured by programming the security bits in the SCR blocks. Each slave has an SCR containing a security bit for each available master, which enables a different security access level to be programmed for each master-slave pair. At reset, all accesses are configured to be secure. When a master-slave security bit is 0, only secure packets are allowed to pass the firewall and reach the intended target. When the master-slave security bit is 1, the firewall passes the transaction.

Table 6-10: Security Bit Value

Secure Flag Value	Access
0	Only secure packets are allowed to pass the firewall.
1	Transaction packets are allowed to pass the firewall.

When a transaction packet is sent from a master, the master also drives a secure flag signal on the bus. This flag indicates whether the attempted transaction from the master is secure or non-secure. For AXI master accesses, this flag is the `~AxPROT[1]` signal.

For L4 ECC, L4 system or L4 DDR master accesses, this flag is the `MSecure[0]` signal.

When the flag signal is driven high, it indicates a secure access. When the flag signal is low, it indicates a non-secure access.

Secure Transaction Example

A successful secure access attempt by a master, depends on the security bit configuration in the Security Control Register and the value of the master secure flag sent by the master.

All SoC transactions are set to secure out of reset with a return of random data when transactions are blocked. If you want an error response returned for blocked transactions, you must set the `error_response` bit in the `global` register of the `noc_fw_ddr_l3_ddr_scr` module at base address 0xFFD13400. If this bit is clear, transactions blocked by the firewall return random data.

Note: Future devices may not support the return of random data and may only support an error response for blocked firewall transactions. For designs that may be ported to future devices, Altera recommends you to set the `error_response` bit in the `global` register.

To initiate a secure transaction:

1. The master drives the address of the slave as well as the master secure flag signal, indicating that the access is intended to be secure.
2. The firewall compares the programmed security bit of the master-slave peripheral pair to the master secure flag signal to determine if the access is valid.
3. If the access is valid, the transaction passes to the next level. If the access is not valid, random data is fed back to the master.

Table 6-11: Peripheral and System Transaction Results

Security Bit in Firewall Register	Master Secure Flag	Transaction Pass/Fail	Transaction Type
0	0	No	None (Blocked transaction)
0	1	Yes	Secure
1	0	Yes	Non-secure
1	1	Yes	Secure

Non-secure Transaction Configuration

When the SoC is released from reset, only secure transactions are allowed through the firewalls. The user must configure any master-slave pairs that are required to be non-secure.

The SCR registers must be accessed in secure mode to configure the firewall access to non-secure.

1. After the SoC is released from reset, the MPU, FPGA2SOC masters or the DAP accesses the required Security Configuration Register (SCR) by executing a secure write. The peripheral firewall SCR registers are within the `soc_noc_fw_l4_per` register group and the system firewall SCR registers are found within the `soc_noc_fw_l4_sys` register group.
2. To configure the slave to allow a non-secure access by a master, set the corresponding security bit to 1.

The configured master is able to successfully execute a non-secure read or write to the configured slave.

HPS-to-FPGA Firewall

The HPS-to-FPGA firewall is used to filter access to all the FPGA, including the HPS-to-FPGA bridge and the lightweight HPS-to-FPGA bridge.

The secure firewall for the HPS-to-FPGA can be disabled by programming the `lwsoc2fpga` register and `soc2fpga` register within the `noc_fw_soc2fpga` group.

On-Chip RAM Firewall

At reset all memories are secure. Out of reset, regions of memories can be configured for non-secure accesses.

The on-chip RAM is divided into six regions, where the granularity of the address boundary is 4 KB. Within each region, a non-secure or shared memory region can be assigned by programming a `base` and `limit` value in the corresponding `regionNaddr` register, with `N` denoting the region number. Each of these regions can be enabled by writing to the `enable` register or setting the corresponding bit in the `enable_set` register.

When an incoming transaction falls within any enabled non-secure regions, the firewall allows both secure and non-secure packets. When the transaction is outside of any enabled regions, the firewall only allows secure packets.

When a transaction packet is sent from a master, the master also drives a secure master flag signal on the bus. This flag indicates whether the attempted transaction from the master is for a secure or non-secure memory region. When the flag signal is driven high, it indicates a secure access. When the flag signal is low it indicates a non-secure access.

On-Chip RAM Transaction Configuration

Because all memories are secure when they exit reset, the on-chip RAM Security Control Register block must be configured by a secure master before non-secure or shared memory accesses are allowed.

1. Configure the `base` and `limit` fields of the `regionNaddr` register, where `N` denotes the region number.
2. Enable the non-secure memory region by setting the corresponding `regionNenable` bit in the `enable_set` register, where `N` denotes the region number.

SDRAM Firewall

Two SDRAM firewalls are implemented to allow a different security policy depending on the master.

One firewall filters accesses from the MPU and the FPGA-to-SDRAM interface. The other firewall filters accesses from all other masters on the HPS and applies the same security policy to both coherent and non-coherent accesses. Both firewalls are logically and physically split within the interconnect.

The SDRAM Firewall supports a minimum region size of 64 KB. Each master is allowed a different number of firewall regions:

- The MPU can have up to four firewall regions.
- The FPGA-to-SDRAM interface can have up to twelve firewall regions.
- Any HPS bus master can have up to eight firewall regions.

Privilege Filter

If a transaction packet has passed the security firewall, it may pass through a privilege filter. The privilege filter only applies to writes. All reads are passed without exception.

The privilege filter decodes the incoming user or privilege accesses and determines whether to pass or fail the transaction. It is separate from the security firewall and transactions generally carry both privilege and secure bits. The privilege filters are configured in the interconnect by executing a read-modify-write to the

l4_priv register or setting individual bits in the l4_set register. If a privilege bit is set, both privilege and user mode transactions are allowed to the slave. If a privilege bit is cleared using the l4_clear register, then only privileged transactions are allowed to the slave.

The following slaves can be programmed using the privilege registers in the interconnect module:

- HPS-to-FPGA bridge
- Lightweight HPS-to-FPGA bridge
- UARTs
- SP Timers
- I2C Modules
- GPIO
- SD/MMC
- QSPI
- EMAC Modules
- SPI
- Secure and non-secure DMA
- USB
- NAND Controller

The following table shows the result of privileged and user master accesses based on the value of the programmed privilege bit for a particular slave.

Note: All read accesses pass regardless of the privilege bit value.

Table 6-12: Privilege Filter Transaction

Read/Write	Privilege Signal	Privilege bit	Transaction Pass/Fail
Read	0	0	Pass
	0	1	Pass
	1	0	Pass
	1	1	Pass
Write	0	0	Fail
	0	1	Pass
	1	0	Pass
	1	1	Pass

Master Security Policy

Each master has an inherent security transaction capability.

Masters accessing slaves can be configured to one of three different security policies:

- Per transaction: The master is capable of generating secure and non-secure transactions.
- Secure: The master only supplies secure transactions.
- Non-secure: The master only generates non-secure transactions.

At reset, all accesses default to secure transactions. The table below details the transaction capability of each master within the SoC. Some masters are only capable of non-secure transactions.

Table 6-13: Master Transaction Capability

Master	Transaction Capability
DMA	Secure/Non-secure
DAP	Secure/Non-secure
USB OTG 0/1	Non-secure
SD/MMC	Non-secure
EMAC0/1/2	Secure/Non-secure
NAND	Non-secure
FPGA-to HPS Bridge	Secure/Non-secure
ETR	Secure/Non-secure
MPU	Secure/Non-secure
FPGA-to-SDRAM	Secure/Non-secure

Security policies are based on secure and privilege attributes. For instance, if CPU0 is configured to access NAND registers in both secure and non-secure mode and CPU0 attempts an access when the core is in secure or non-secure mode, no error occurs. However, if CPU0 is allowed to access NAND registers only in secure mode and CPU0 is operating in non-secure mode, then CPU0 receives an error response when accessing the NAND registers. If both the security firewall and privilege firewall are implemented, security firewall filters all of the accesses. If an access fails, random data or an error response is sent to the master, depending on how the `error_response` bit in the `global` register of the `noc_fw_ddr_l3_ddr_scr` module is programmed. If access is granted by the security firewall, then the transaction enters the privilege firewall. If access is granted, the request enters the peripheral IP.

Note: Future devices may not support the return of random data and may only support an error response for blocked firewall transactions. For designs that may be ported to future devices, Altera recommends you to set the `error_response` bit in the `global` register.

Secure Interconnect Reset

When the SoC is released from reset, every slave on the interconnect is in the secure state (boot secure).

The only masters capable of secure accesses out of reset are:

- MPU
- DAP
- HPS-to-FPGA fabric

Revision History

The table below holds the document revision history for *SoC Security* Chapter.

Date	Version	Changes
October 2016	2016.10.28	Maintenance release
May 2016	2016.05.27	Maintenance release
May 2016	2016.05.03	• Added note regarding blocked firewall transaction responses in the "Secure Firewall" section • Updated content and added figures to "JTAG" section
November 2015	2015.11.02	Clarified initialization steps in "Secure Initialization Overview" section
May 2015	2015.05.04	Added Address Map and Register information.
December 2014	2014.12.15	• Added "Secure Fuses" section under "Secure Initialization" section. • Added summary of "Security State" and "Security Check" and "Secure Serial Interface" features. • Added "MPU" and "JTAG" sub-sections to the "Secure Debug" section. • Added information regarding CSEL programming in the "Clock Configuration" section. • Added "FPGA Security Features" section
August 2014	2014.08.18	Initial Release

System Interconnect 7

2016.10.28

a10_5v4 Subscribe Send Feedback

The components of the hard processor system (HPS) communicate with one another, and with other portions of the SoC device, through the system interconnect. The system interconnect consists of the following blocks:

- The main level 3 (L3) interconnect
- The SDRAM L3 interconnect
- The level 4 (L4) buses

The system interconnect is the main communication bus for the MPU and all IPs in the SoC device.

The system interconnect is implemented by the Arteris FlexNoC network-on-chip (NoC) interconnect module.

The system interconnect supports the following features:

- An L3 interconnect that provides high-bandwidth routing between masters and slaves in the HPS
- SDRAM L3 interconnect, providing access to a hard memory controller in the FPGA fabric through a multiport front end (MPFE) scheduler for sharing the external SDRAM between multiple masters in the SoC device
- Independent L4 buses running in several clock domains. These buses handle data traffic for low- to mid-level bandwidth slave peripherals and accesses to peripheral control and status registers throughout the address map.
- On-chip debugging and tracing capabilities
- Security firewalls with the following capabilities:
  - Secure or nonsecure access configured per peripheral
  - Privileged or user access configured per peripheral (for some peripherals)
  - Security optionally configured per transaction at the master
- Quality of service (QoS) with three programmable levels of service on a per-master basis

Related Information

www.arteris.com
For information about the FlexNoC Network-on-Chip Interconnect, refer to the Arteris website.

About the System Interconnect

Features of the System Interconnect

The system interconnect supports high-throughput peripheral devices. The system interconnect has the following characteristics:

- Byte oriented address handling
- Data bus width up to 128 bits
- ARM TrustZone-compliant firewall and security support, with additional security features configurable per master
- Programmable quality-of-service (QoS) optimization
- Dedicated SDRAM L3 interconnect, providing access and scheduling for the hard memory controller in the FPGA portion of the SoC device
- On-chip debug and tracing capabilities
- Multiple independent L4 buses with independent clock domains and protocols

  The L4 buses are each connected to a master in the main L3 interconnect for control and status register access. Each L4 bus is 32 bits wide and is connected to multiple slaves. The L4 buses also provide a low- to mid-level tier of the system interconnect for lower-bandwidth slave peripherals. Each L4 bus is 32 bits wide and operates in a separate clock domain.

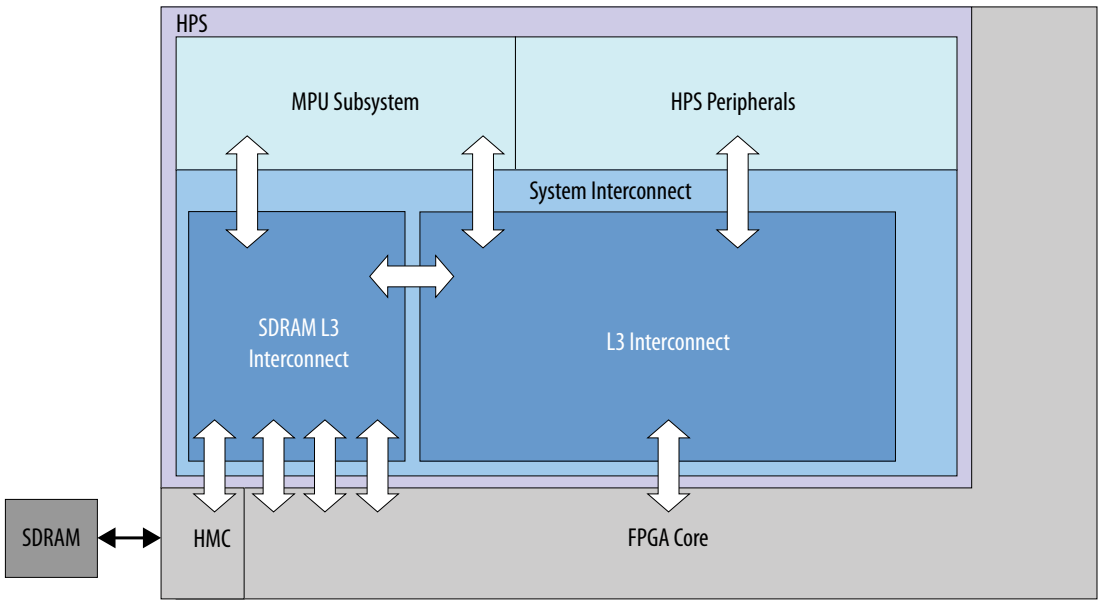
System Interconnect Block Diagram and System Integration

System Interconnect Block Diagram

The following figures show the system interconnect, including the main L3 interconnect, SDRAM L3 interconnect, and L4 buses.

Figure 7-1: High-Level System Interconnect Block Diagram

The following figure shows the relationships among the system interconnect and other major SoC components.



Related Information

Arria 10 HPS Master-to-Slave Connectivity Matrix on page 7-3

Connectivity

Arria 10 HPS Master-to-Slave Connectivity Matrix

The system interconnect is a highly efficient packet-switch network.

The following figure shows the connectivity of all the master and slave interfaces in the system interconnect.

Figure 7-2: Master-to-Slave Connectivity

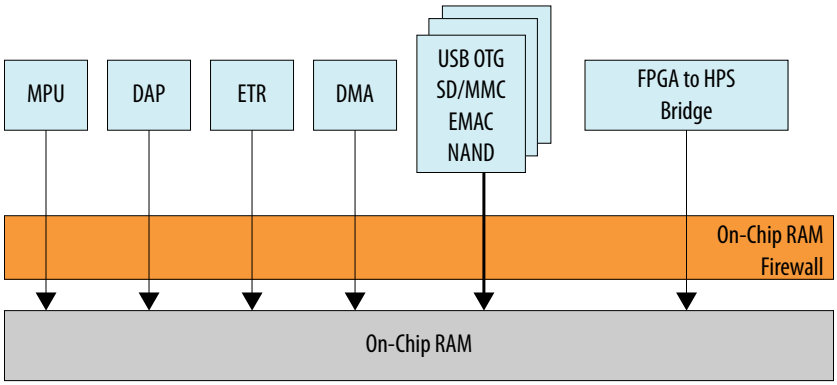
Masters	Slaves														
	L4 Main Bus Slaves	L4 MP Bus Slaves	L4 AHB Bus Slaves	L4 SP Bus Slaves	L4 SYS Bus Slaves	L4 ECC Bus Slaves	DAP	STM	On-Chip RAM	Boot ROM	ACP	DDR	Lightweight HPS-to-FPGA Bridge	HPS-to-FPGA Bridge	Service Network
MPU L2 Cache Master 0	✓	✓	✓	✓	✓		✓	✓	✓	✓			✓	✓	✓
MPU L2 Cache Master 1												✓			
FPGA-to-HPS Bridge	✓	✓	✓	✓	✓(1)	✓	✓	✓	✓		✓	✓			✓
FPGA-to-SDRAM Interface												✓			
DMA	✓	✓	✓	✓	✓		✓	✓	✓		✓	✓	✓	✓	
EMAC 0/1/2									✓		✓	✓	✓	✓	
USB 0/1									✓		✓	✓	✓	✓	
NAND									✓		✓	✓	✓	✓	
SD/MMC									✓		✓	✓	✓	✓	
ETR							✓		✓			✓	✓	✓	
DAP	✓	✓	✓	✓	✓	✓	✓		✓	✓	✓	✓	✓	✓	✓

Note:
1. The FPGA-to-HPS Bridge is connected to all L4 system slaves except the FPGA manager.

On-Chip RAM

Figure 7-3: On-Chip RAM Connections

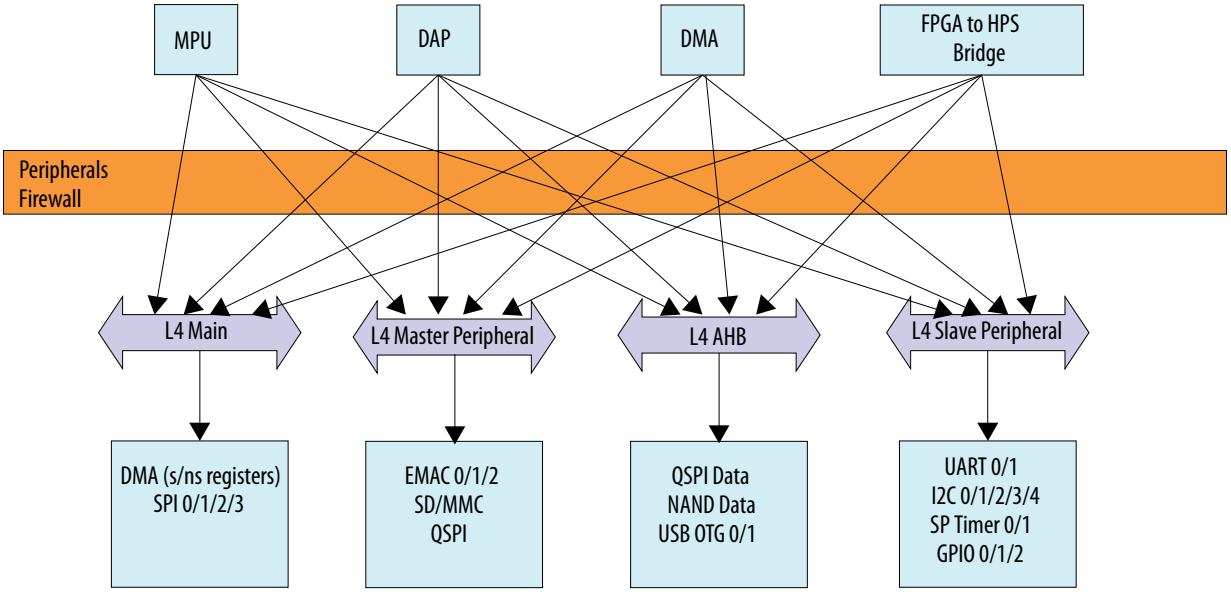
HPS masters connecting to the on-chip RAM



Peripherals Connections

Figure 7-4: Peripherals Connections

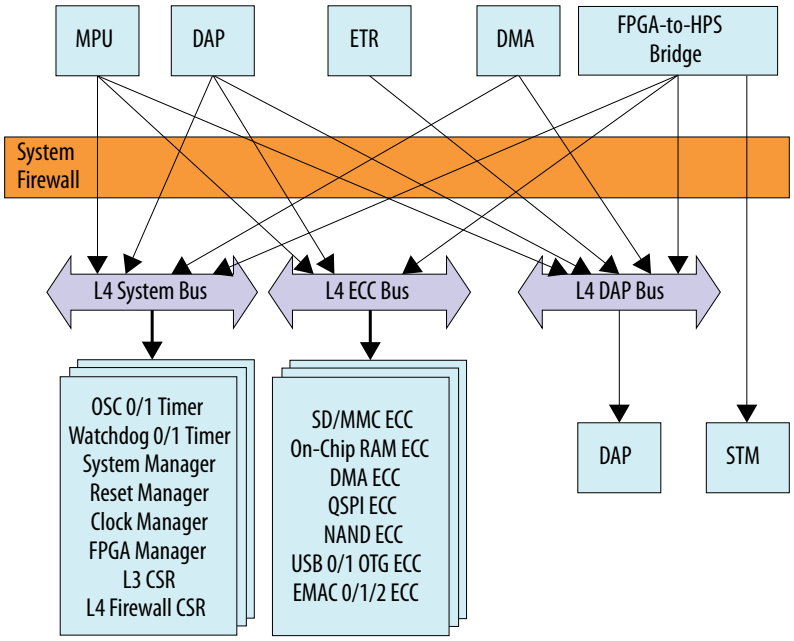
HPS masters connecting to HPS peripherals



System Connections

Figure 7-5: System Connections

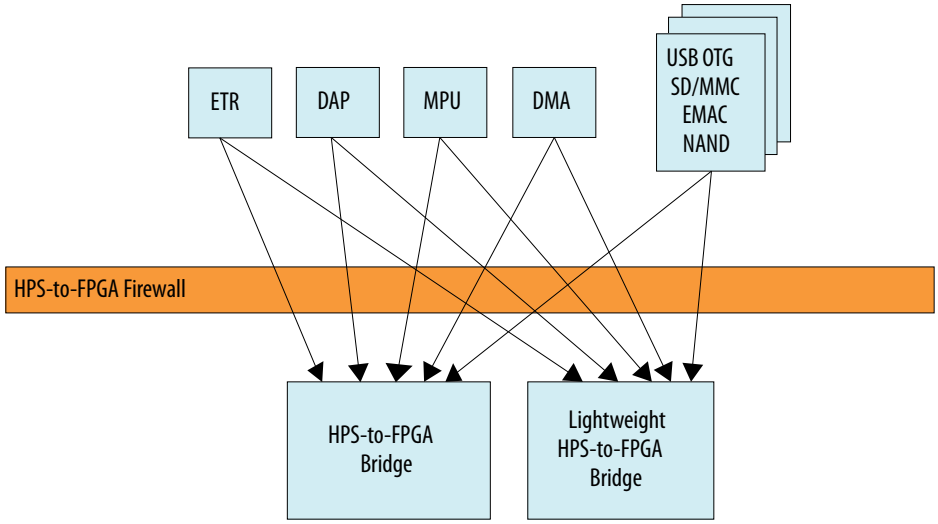
HPS masters connecting to HPS system peripherals



HPS-to-FPGA Bridge

Figure 7-6: HPS-to-FPGA Bridge Connections

HPS masters connecting to the HPS-to-FPGA bridges



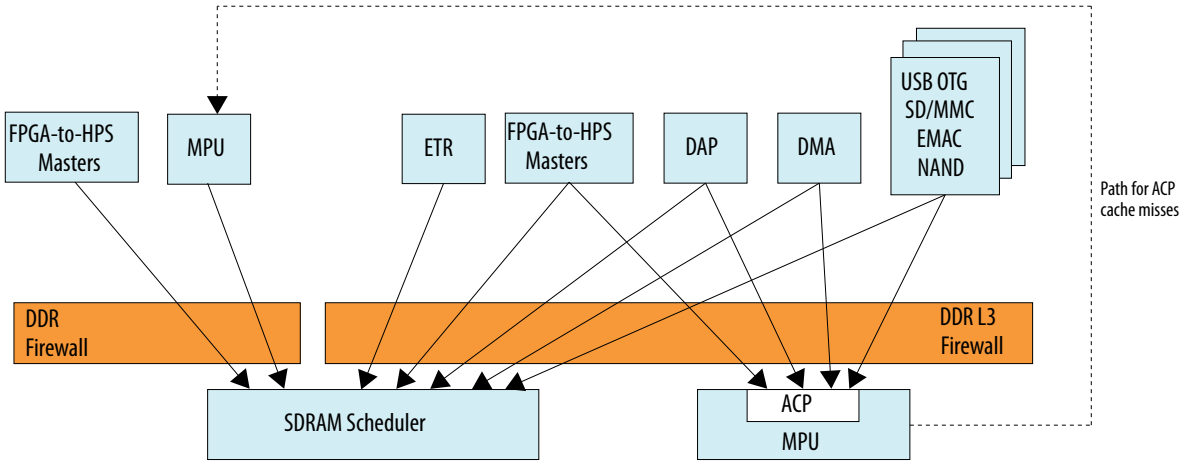
Related Information

HPS-FPGA Bridges on page 8-1
For more information, refer to the *HPS-FPGA Bridges* chapter.

SDRAM Connections

Figure 7-7: SDRAM Connections

HPS masters connecting to the SDRAM scheduler. Note that cache misses to the MPU Accelerator Coherency Port (ACP) pass through a secondary firewall.



Related Information

SDRAM L3 Interconnect Block Diagram and System Integration on page 7-9
More detailed information about the SDRAM L3 interconnect

System Interconnect Architecture

The system interconnect has a transaction-based architecture that functions as a partially-connected fabric. Not all masters can access all slaves.

The system interconnect is a network on chip (NoC) providing interface widths up to 128 bits, connecting to the L4 slave buses and to HPS and FPGA masters and slaves.

The system interconnect provides low-latency connectivity to the following bridges:

- HPS-to-FPGA bridge
- Lightweight HPS-to-FPGA bridge
- FPGA-to-HPS bridge
- Three FPGA-to-SDRAM ports

SDRAM L3 Interconnect Architecture

The SDRAM L3 interconnect provides access to the hard memory controller in the FPGA portion of the SoC device.

The SDRAM L3 interconnect is part of the system interconnect, and includes these components:

- SDRAM scheduler
- SDRAM adapter

The SDRAM L3 interconnect operates in a clock domain that is 1/2 the speed of the hard memory controller clock.

Arria 10 HPS Secure Firewalls

You can use the system interconnect firewalls to enforce security policies for slave and memory regions in the system interconnect.

The firewalls support the following features:

- Secure or non-secure access configurable per peripheral
- Privileged or user access configurable for some peripherals
- Per-transaction security

Each firewall contains the security configuration registers (SCRs) that set security policies and define which transactions are allowed to go through the firewall. If you want to generate an error any time a transaction is blocked in the firewall, then you must set the `error_response` bit in the `global` register of the `noc_fw_ddr_l3_ddr_scr` module at base address 0xFFD13400. If this bit is clear, transactions blocked by the firewall return random data.

Note: Future devices may not support the return of random data and may only support an error response for blocked firewall transactions. For designs that may be ported to future devices, Altera recommends you to set the `error_response` bit in the `global` register.

There are five main firewalls in the HPS:

- Peripheral
- System
- HPS-to-FPGA
- On-Chip RAM
- SDRAM (which includes DDR and DDR L3 firewalls)

Related Information

- **System Interconnect** on page 7-1
  For more information about the System Interconnect, refer to the *Interconnect* chapter.
- **SoC Security** on page 6-1

About the Rate Adapters

The L3 interconnect contains a rate adapter wherever a low-bandwidth channel transfers data to a high-bandwidth channel.

Related Information

Functional Description of the Rate Adapters on page 7-26

About the SDRAM L3 Interconnect

The hard processor system (HPS) provides a specialized SDRAM L3 Interconnect dedicated to SDRAM accesses.

The SDRAM L3 Interconnect contains an SDRAM scheduler and an SDRAM adapter. The SDRAM scheduler functions as a multi-port front end (MPFE) for multiple HPS masters. The SDRAM adapter is responsible for connecting the HPS to the SDRAM hard memory controller in the FPGA portion of the device. The SDRAM L3 interconnect is part of the system interconnect.

Features of the SDRAM L3 Interconnect

The SDRAM L3 interconnect supports the following features:

- Connectivity to the SDRAM hard memory controller supporting:
  - DDR4-SDRAM
  - DDR3-SDRAM
- Integrated SDRAM scheduler, functioning as a multi-port front end (MPFE)
- Configurable SDRAM device data widths
  - 16-bit, with or without 8-bit error-correcting code (ECC)
  - 32-bit, with or without 8-bit ECC
  - 64-bit, with or without 8-bit ECC
- High-performance ports
  - MPU subsystem port
  - Main L3 interconnect port
  - Three FPGA ports
    - Two 32-, 64-, or 128-bit ports
    - One 32- or 64-bit port
  - Firewall and security support port

Note: At system startup, the SDRAM I/O pins can be configured separately from the FPGA fabric, allowing the SoC HPS to boot before any soft logic is configured in the FPGA.

Related Information

- **Boot Source I/O Pins** on page 30-8
  Details of the SoC hardware and software boot flow
- **Configuring HPS I/O Multiplexing** on page 25-8
  More information about configuring I/O pins

SDRAM L3 Interconnect Block Diagram and System Integration

The SDRAM L3 interconnect is composed of two main blocks: SDRAM adapter and SDRAM scheduler.

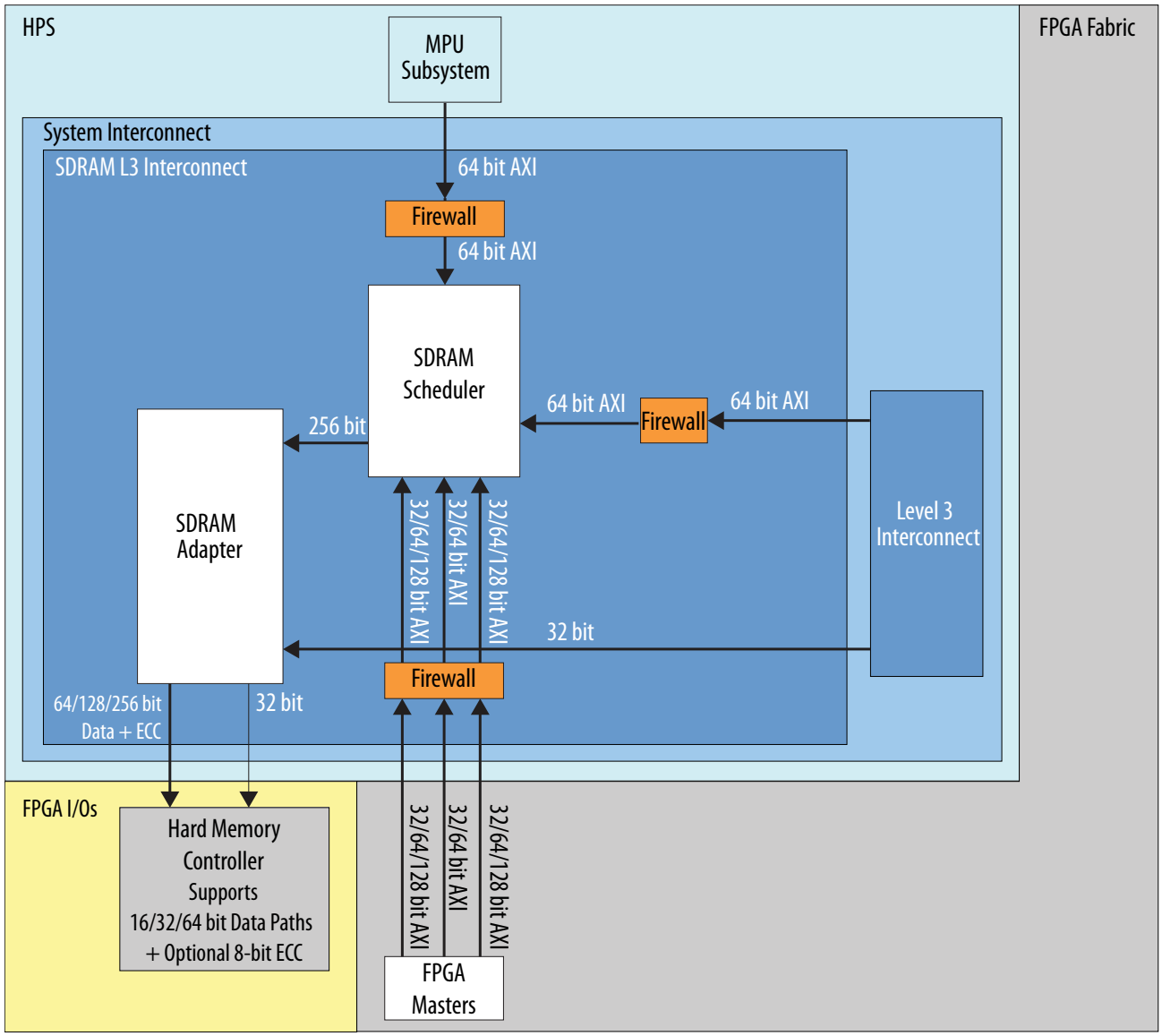
The SDRAM adapter is responsible for bridging the hard memory controller in the FPGA fabric to the SDRAM scheduler. The adapter is also responsible for ECC generation and checking.

The ECC register interface provides control to perform memory and ECC logic diagnostics.

The SDRAM scheduler is an MPFE, responsible for arbitrating collisions and optimizing performance in traffic to the SDRAM controller in the FPGA portion of the device.

Three ARM Advanced Microcontroller Bus Architecture (AMBA) Advanced eXtensible Interface (AXI) ports are exposed to the FPGA fabric, allowing soft logic masters to access the SDRAM controller through the same scheduler unit as the MPU subsystem and other masters within the HPS. The MPU has access to the SDRAM adapter's control interface to the hard memory controller.

Figure 7-8: SDRAM L3 Interconnect Block Diagram



The hard memory controller in the FPGA portion of the device has a dedicated connection to the SDRAM L3 interconnecct. This connection allows the hard memory controller to become operational before the rest of the FPGA has been configured.

About the SDRAM Scheduler

The SDRAM scheduler functions as a multi-port front end (MPFE), scheduling transactions from multiple masters to the SDRAM.

The SDRAM scheduler supports the following masters:

- The MPU
- The L3 system interconnect
- The FPGA-to-SDRAM bridges

The SDRAM scheduler arbitrates among transactions initiated by the masters, and determines the order of operations. The scheduler arbitrates among the masters, ensuring optimal interconnect performance based on configurable quality-of-service settings.

The SDRAM scheduler can be configured through the registers.

Related Information

- **Functional Description of the QoS Generators** on page 7-34
  Detailed description of the system interconnect's quality-of-service features
- **Quality of Service in the SDRAM Scheduler** on page 7-30

About Quality of Service and Arbitration

The system interconnect supports quality-of-service (QoS) optimization through programmable QoS generators. The QoS generators are located on interconnect initiators, which correspond to master interfaces. Each QoS generator creates control signals that prioritize the handling of individual transactions to meet performance requirements.

The purpose of controlling QoS is to prevent one initiator from using up the interconnect's bandwidth at the expense of other initiators.

Proper QoS settings depend on your performance requirements for each component and peripheral, and for system performance as a whole. Altera recommends that you become familiar with QoS optimization techniques before you try to change the QoS settings in the HPS system interconnect.

Related Information

- **Arria 10 HPS Master-to-Slave Connectivity Matrix** on page 7-3
- **Functional Description of the QoS Generators** on page 7-34

About the Service Network

The service network is physically separate from the NoC datapath.

Through the service network, you can perform these tasks:

- Access internal interconnect registers
- Update master and slave peripheral security features

About the Observation Network

The observation network connects probes to the CoreSight trace funnel through the debug channel. It is physically separate from the NoC datapath.

The observation network connects probes to the observer, which is a port in the CoreSight trace funnel. Through the observation network, you can perform these tasks:

- Enable error logging
- Selectively trace transactions in the system interconnect
- Collect HPS transaction statistics and profiling data

The observation network consists of probes in key locations in the interconnect, plus connections to observers. The observation network works across multiple clock domains, and implements its own clock crossing and pipelining where needed.

The observation network sends probe data to the CoreSight subsystem through the ATB interface. Software can enable probes and retrieve probe data through the interconnect observation registers.

Related Information

Functional Description of the Observation Network on page 7-40

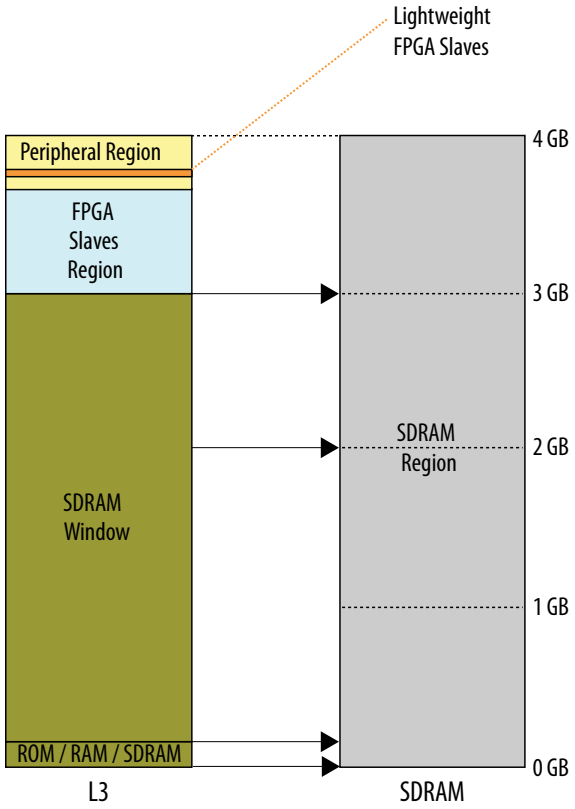
Functional Description of the System Interconnect

The system interconnect provides access to a 4 GB address space.

Address spaces are divided into one or more nonoverlapping contiguous regions.

Figure 7-9: HPS Address Space Relationships

Positions of HPS address spaces. For readability, some memory regions are shown out-of-scale to their addresses.



The window regions provide access to other address spaces. The thin black arrows indicate which address space is accessed by a window region (arrows point to accessed address space).

The following table shows the base address and size of each region that is common to the L3 and MPU address spaces.

Table 7-1: Common Address Space Regions

Region Name	Description	Base Address	Size
FPGASLAVES	FPGA slaves	0xC0000000	960 MB
PERIPH	Peripheral	0xFC000000	64 MB
LWFPGASLAVES [8]	Lightweight FPGA slaves	0xFF200000	2 MB

Related Information

Arria 10 HPS Available Address Maps on page 7-13
Details of the L3 and MPU address maps

System Interconnect Address Spaces

The system interconnect supports multiple address spaces.

Each address space uses some or all of the 4 GB address range. The address spaces overlap. Depending on the configuration, different address spaces are visible in different regions for each master.

The following address spaces are available:

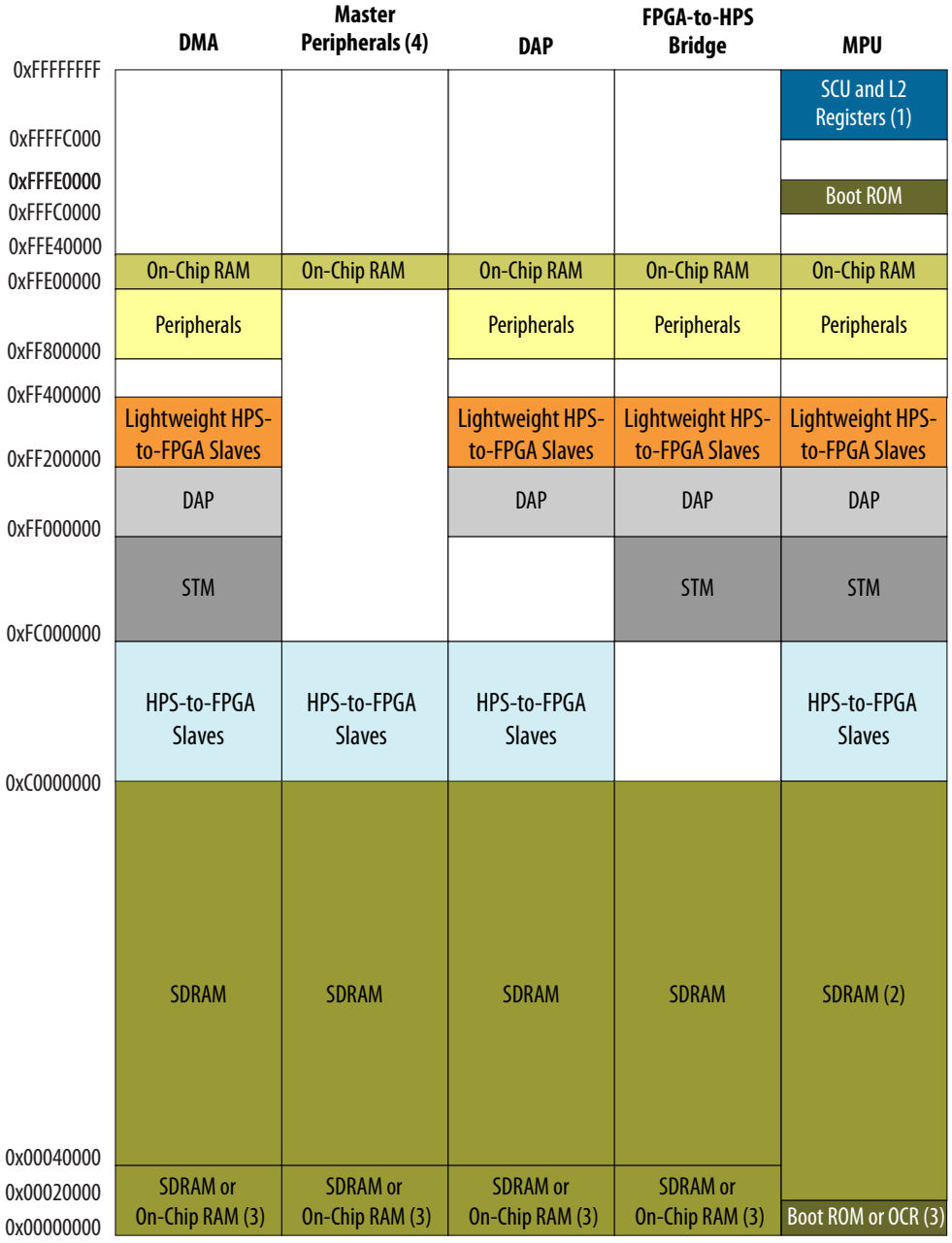
- The L3 address space
- The MPU address space
- The SDRAM address space

Arria 10 HPS Available Address Maps

The following figure shows the default system interconnect address maps for all masters. The figure is not to scale.

[8] The LWFPGASLAVES region is part of the "PERIPH" region.

Figure 7-10: Address Maps for System Interconnect Masters



Notes on Address Maps

(1) Transactions on these addresses are directly decoded by the SCU and L2 cache.

(2) The MPU accesses SDRAM through a dedicated port to the SDRAM scheduler. The SDRAM window size also depends on L2 cache filtering.

(3) This region can be configured to access on-chip RAM, by using the `noc_addr_remap_value`, `noc_addr_remap_set`, and `noc_addr_remap_clear` registers, in the system manager.

(4) The following peripherals can master the interconnect:

- Ethernet MACs
- USB-2.0 OTG controllers
- NAND controller
- ETR
- SD/MMC controller

For the MPU L3 master, either the boot ROM or on-chip RAM can map to address 0x0 and obscure the lowest 128 KB or 256 KB of SDRAM.

At boot time, the MPU does not have access to the SDRAM address space from the top of ROM or on-chip RAM to 0x00100000. This is because the MPU's SDRAM access is controlled by the MPU L2 filter registers, which only have a granularity of 1 MB. After booting completes, the MPU can change address filtering to use the lowest 1 MB of SDRAM.

For non-MPU masters, either the on-chip RAM or the SDRAM maps to address 0x0. When mapped to address 0x0, the on-chip RAM obscures the lowest 256 KB of SDRAM for non-MPU masters.

Related Information

- **Memory Map Remap Bits** on page 7-20
  Details of how to map ROM and RAM to 0x00000000
- **Cortex-A9 MPU Subsystem Register Implementation** on page 9-46
  Information about HPU register implementation, including the SCU and L2 cache registers
- **Address Remapping** on page 7-19
  Information about controlling system interconnect memory maps
- **SDRAM Address Space** on page 1-18
  An overview of the SDRAM address space is in the *Introduction to the Hard Processor System* chapter.
- **The SDRAM Region** on page 9-11
  More information about L2 cache filtering
- **System Manager** on page 5-1
  For more information about setting registers in the system manager, refer to the *System Manager* chapter.

Arria 10 HPS L3 Address Space

The Arria 10 HPS L3 address space is 4 GB and applies to all L3 masters except the MPU subsystem.

The system interconnect `noc_addr_remap_value`, `noc_addr_remap_set`, and `noc_addr_remap_clear` registers, in the System Manager, determine if the address space starting at address 0x0 is mapped to the on-chip RAM (256 KB) or the SDRAM. SDRAM is mapped to address 0x0 on reset.

The L3 address space configurations contain the regions shown in the following table:

Table 7-2: L3 Address Space Regions

Description	Condition	Base Address	End Address	Size
SDRAM window (without on-chip RAM)	`remap0` set, `remap1` clear (9)	0x00000000	0xBFFFFFFF	3 GB

(9) For details about the `noc_addr_remap_value` register, refer to "Memory Map Remap Bits"

Description	Condition	Base Address	End Address	Size
On-chip RAM (low mapping)	`remap0` set, `remap1` set (9)	0x00000000	0x0003FFFF	256 KB
SDRAM window (with on-chip RAM)	`remap0` set, `remap1` set (9)	0x00040000	0xBFFFFFFF	3145471 KB = 3 GB - 256 KB
HPS-to-FPGA	Not visible to FPGA-to-HPS bridge.	0xC0000000	0xFBFFFFFF	960 MB
System trace macrocell	Always visible to DMA and FPGA-to-HPS	0xFC000000	0xFEFFFFFF	48 KB
Debug access port	Not visible to master peripherals. Always visible to other masters.	0xFF000000	0xFF1FFFFF	2 MB
Lightweight HPS-to-FPGA	Not visible to master peripherals. Always visible to other masters.	0xFF200000	0xFF3FFFFF	2 MB
Peripherals	Not visible to master peripherals. Always visible to other masters.	0xFF800000	0xFFFDFFFF	8064 KB
On-chip RAM (high mapping)	Always visible	0xFFE00000	0xFFFE3FFF	256 KB

The boot ROM and internal MPU registers (SCU and L2) are not accessible to L3 masters.

Cache coherent memory accesses have the same view of memory as the MPU.

SDRAM Window Region

The SDRAM window region is 3 GB and provides access to the bottom 3 GB of the SDRAM address space.

On-Chip RAM Region, Low Mapping

The system interconnect `noc_addr_remap_value` register determines if the 256 KB starting at address 0x0 is mapped to the on-chip RAM or the SDRAM. The SDRAM is mapped to address 0x0 on reset.

HPS-to-FPGA Slaves Region

The HPS-to-FPGA slaves region provides access to 960 MB of slaves in the FPGA fabric through the HPS-to-FPGA bridge.

Lightweight HPS-to-FPGA Slaves Region

The lightweight HPS-to-FPGA slaves provide access to slaves in the FPGA fabric through the lightweight HPS-to-FPGA bridge.

Peripherals Region

The peripherals region includes slaves connected to the L3 interconnect and L4 buses.

On-Chip RAM Region, High Mapping

The on-chip RAM is always mapped (independent of the boot region contents).

Related Information

- **Memory Map Remap Bits** on page 7-20
  Details of how to map ROM and RAM to 0x00000000
- **Arria 10 HPS Available Address Maps** on page 7-13
  Details of the L3 and MPU address maps
- **HPS Address Spaces** on page 1-17
  More information about L3 address space mapping is in the *Introduction to the Hard Processor System* chapter.
- **Cortex-A9 Microprocessor Unit Subsystem** on page 9-1
  For general information about the MPU subsystem, refer to the *Cortex-A9 Microprocessor Unit Subsystem* chapter.

MPU Address Space

The MPU address space is 4 GB and applies to MPU masters.

Addresses generated by the MPU are decoded in three ways:

- By default, MPU accesses to locations between 0x100000 (1 MB) to 0xC0000000 (3 GB) are made to the SDRAM controller.
- Addresses in the SCU and L2 register region (0xFFFFC000 to 0xFFFFFFFF) are the SCU and L2 bus.
- Accesses to all other locations are made to the L3 interconnect.

The MPU L2 cache controller contains a master connected to the main L3 interconnect and a master connected to the SDRAM L3 interconnect.

The system interconnect `noc_addr_remap_value`, `noc_addr_remap_set`, and `noc_addr_remap_clear` registers, in the System Manager, determine if the address space starting at address 0x0 is mapped to the on-chip RAM (256 KB) or the ROM (128 KB). The ROM is mapped to address 0x0 on reset.

The MPU address space contains the following regions:

Table 7-3: MPU Default Address Space Regions

Description	Condition	Base Address	End Address	Size
Boot ROM	`remap0` clear, `remap1` clear (10)	0x00000000	0x0001FFFF	128 KB
SDRAM window	Always visible	0x00100000	0xBFFFFFFF	3071 MB (3 GB – 1 MB)
HPS-to-FPGA	Always visible	0xC0000000	0xFBFFFFFF	960 MB (1 GB – 64 MB)

(10) For details about the `noc_addr_remap_value` register, refer to "Memory Map Remap Bits".

Description	Condition	Base Address	End Address	Size
System trace macrocell	Always visible	0xFC000000	0xFFEFFFFF	48 KB
Debug access port	Always visible	0xFF000000	0xFF1FFFFF	2 MB
Lightweight HPS-to-FPGA	Always visible	0xFF200000	0xFF3FFFFF	2 MB
Peripherals	Always visible	0xFF800000	0xFFDFFFFF	6 MB
On-chip RAM	Always visible	0xFFE00000	0xFFFE3FFF	256 KB
Boot ROM	Always visible	0xFFFC0000	0xFFFDFFFF	128 KB
SCU and L2 registers	Always visible	0xFFFFC000	0xFFFFFFFF	16 KB

Boot Region

The boot region is 1 MB, based at address 0x0. The boot region is visible to the MPU only when the L2 address filter start register is set to 0x100000. The L3 `noc_addr_remap_value` control register determines if the boot region is mapped to the on-chip RAM or the boot ROM.

The boot region is mapped to the boot ROM on reset. Only the lowest 128 KB of the boot region are legal addresses because the boot ROM is only 128 KB.

When the L2 address filter start register is set to 0, SDRAM obscures access to the boot region. This technique can be used to gain access to the lowest SDRAM addresses after booting completes.

SDRAM Window Region

The SDRAM window region provides access to a large, configurable portion of the 4 GB SDRAM address space. The address filtering start and end registers in the L2 cache controller define the SDRAM window boundaries.

The boundaries are megabyte-aligned. Addresses within the boundaries route to the SDRAM master, while addresses outside the boundaries route to the system interconnect master.

Addresses in the SDRAM window match addresses in the SDRAM address space. Thus, the lowest 1 MB of the SDRAM is not visible to the MPU unless the L2 address filter start register is set to 0.

HPS-to-FPGA Slaves Region

The HPS-to-FPGA slaves region provides access to slaves in the FPGA fabric through the HPS-to-FPGA bridge. Software can move the top of the SDRAM window by writing to the L2 address filter end register. If higher addresses are made available in the SDRAM window, part of the FPGA slaves region might be inaccessible to the MPU.

Lightweight HPS-to-FPGA Slaves Region

The lightweight FPGA slaves provide access to slaves in the FPGA fabric through the lightweight HPS-to-FPGA bridge.

Peripherals Region

The peripherals region is near the top of the address space. The peripherals region includes slaves connected to the L3 interconnect and L4 buses.

On-Chip RAM Region

The on-chip RAM is always mapped near the top of the address space, independent of the boot region contents.

Boot ROM Region

The boot ROM is always mapped near the top of the address space, independent of the boot region contents.

SCU and L2 Registers Region

The SCU and L2 registers region provides access to internally-decoded MPU registers (SCU and L2).

SDRAM Address Space

The SDRAM address space is up to 4 GB. The entire address space can be accessed through the FPGA-to-SDRAM interface from the FPGA fabric. The total amount of SDRAM addressable from the other address spaces varies.

There are cacheable and non-cacheable views into the SDRAM space. When a master of the system interconnect performs a cacheable access to the SDRAM, the transaction is performed through the ACP port of the MPU subsystem. When a master of the system interconnect performs a non-cacheable access to the SDRAM, the transaction is performed through the 32-bit main L3 interconnect master of the SDRAM L3 interconnect.

Address Remapping

The system interconnect supports address remapping through the `noc_addr_remap_value`, `noc_addr_remap_set`, and `noc_addr_remap_clear` registers in the system manager. Remapping allows software to control which memory device (on-chip RAM or boot ROM) is accessible at address 0x0. The remap registers can be modified by the MPU and the FPGA-to-HPS bridge.

The following master interfaces are affected by the remap bits:

- MPU master interface
  - L2 cache master 0 interface
- Non-MPU master interfaces
  - DMA master interface
  - Master peripheral interfaces
  - Debug Access Port (DAP) master interface
  - FPGA-to-HPS bridge master interface

Related Information

- **Cortex-A9 Microprocessor Unit Subsystem** on page 9-1
  For general information about the MPU subsystem, refer to the *Cortex-A9 Microprocessor Unit Subsystem* chapter.

- **System Manager** on page 5-1
  For more information about setting registers in the system manager, refer to the *System Manager* chapter.

Memory Map Remap Bits

Table 7-4: Remap Bit Usage

remap0	remap1	0x00000000 (MPU Master Interface)	0x00000000 (Non-MPU Master Interfaces)	0xFFE00000 (All Maps)	0xFFFC0000 (MPU & DAP)
0	0	Boot ROM	SDRAM	On-Chip RAM	Boot ROM
0	1	Invalid setting			
1	0	SDRAM	SDRAM	On-Chip RAM	Boot ROM
1	1	On-Chip RAM	On-Chip RAM	On-Chip RAM	Boot ROM

L2 filter registers in the MPU subsystem, not the interconnect, allow the SDRAM to be remapped to address `0x0` for the MPU.

Secure Transaction Protection

The system interconnect provides two levels of secure transaction protection:

- Security firewalls—Enforce secure read and write transactions.
- Privilege filter—Leverages the firewall mechanism and provides additional security by filtering the privilege level of L4 slave transactions. The privilege filter applies to writes only.

All slaves on the SoC are placed behind a security firewall. A subset of slaves are also placed behind a privilege filter. Transactions to these slaves must pass both a security firewall and the privilege filter to reach the slave.

Related Information

Functional Description of the Firewalls on page 7-27

System Interconnect Master Properties

The system interconnect connects to slave interfaces through the main L3 interconnect and SDRAM L3 interconnect.

Table 7-5: System Interconnect Master Interfaces

Master	Interface Width	Clock	Security	SCR(11) Access	Privilege	Issuance (Read/ Write/ Total)	Type
MPU Subsystem L2 cache M0/1	64	mpu_l2ram_ clk	Per Transaction	Yes	Transaction based	7/12/23	AXI

(11) Security control register (SCR)

Master	Interface Width	Clock	Security	SCR[11] Access	Privilege	Issuance (Read/ Write/ Total)	Type
FPGA-to-HPS Bridge [12]	32/64/128	fpga2hps_clk	Per Transaction	Yes	Transaction based	8/8/8	AXI
FPGA-to-SDRAM Ports [12]	32/64/128	f2h_sdram_clk[2:0]	Per Transaction	No	Transaction based	8/8/8	AXI
DMA	64	l4_main_clk	Per Transaction	No	User mode	8/8/8	AXI
EMAC 0/1/2	32	l4_mp_clk	Secure/Non-Secure	No	User mode	16/16/32	AXI
USB OTG 0/1	32	l4_mp_clk	Nonsecure	No	User mode	2/2/4	AHB
NAND	32	l4_mp_clk	Nonsecure	No	User mode	1/1/2	AXI
SD/MMC	32	l4_mp_clk	Nonsecure	No	User mode	2/2/4	AHB
ETR	32	cs_at_clk	Per Transaction	No	Transaction based	32/1/32	AXI
AHB-AP	32	l4_mp_clk	Per Transaction	Yes	Transaction based	1/1/1	AHB

Related Information

SoC Security on page 6-1
For more information about master/slave security, refer to the *SoC Security* chapter.

Master Caching and Buffering Overrides

Some of the peripheral masters connected to the system interconnect do not have the ability to drive the caching and buffering signals of their interfaces. The system manager provides registers so that you can enable cacheable and bufferable transactions for these masters. The system manager drives the caching and buffering signals of the following masters:

Master Peripheral	System Manager Registers
EMAC0, EMAC1, and EMAC2	`emac0`, `emac1`, and `emac2`
USB OTG 0 and USB OTG 1	`usb0_l3master` and `usb1_l3master`
NAND flash	`nand_l3master`
SD/MMC	`sdmmc_l3master`

[11] Security control register (SCR)
[12] Ensure that Avalon-MM burst transactions into the HPS do not cross the 4 KB address boundary restriction specified by the AXI protocol.

At reset time, the system manager drives the cache and buffering signals for these masters low. In other words, the masters listed do not support cacheable or bufferable accesses until you enable them after reset. There is no synchronization between the system manager and the system interconnect, so avoid changing these settings when any of the masters are active.

Related Information

System Manager on page 5-1
For more information about enabling or disabling this feature, refer to the *System Manager* chapter.

System Interconnect Slave Properties

The system interconnect connects to various slave interfaces through the main L3 interconnect, the SDRAM L3 interconnect, and the L4 peripheral buses. After reset, all slave interfaces are set to the secure state.

Table 7-6: System Interconnect Slave Interfaces

Slave	Interface Width	Clock	Acceptance (Read/ Write/Total) (13)	Security	Privilege	Interface Type
SP Timer 0/1/2/3	32	l4_sp_clk	1/1/1	Boot Secure (14)	User Mode	APB(15)
I2C 0/1/2/3/4	32	l4_sp_clk	1/1/1	Boot Secure	User Mode	APB (15)
UART 0/1	32	l4_sp_clk	1/1/1	Boot Secure	User Mode	APB (15)
GPIO 0/1/2	32	l4_sp_clk	1/1/1	Boot Secure	User Mode	APB(15)
SD/MMC CSR	32	l4_mp_clk	1/1/1	Boot Secure	User Mode	APB(15)
EMAC 0/1/2	32	l4_mp_clk	1/1/1	Boot Secure	User Mode	APB(15)
OSC Timer 0/1/2/3	32	l4_sys_free_clk	1/1/1	Secure (16)	Privileged	OCP
Watchdog 0/1/2/3	32	l4_sys_free_clk	1/1/1	Secure	Privileged	APB(15)
Clock Manager	32	l4_sys_free_clk	1/1/1	Secure	Privileged	OCP
Reset Manager	32	l4_sys_free_clk	1/1/1	Secure	Privileged	OCP

(13) Acceptance is the maximum number of transactions accepted.
(14) "Boot Secure" means the slave is in the secure state after reset.
(15) The Advanced Peripheral Bus (APB™) slave does not support byte enables. All writes must be 32 bits wide.
(16) "Secure" means that the slave is permanently in the secure state.

Slave	Interface Width	Clock	Acceptance (Read/ Write/Total) (13)	Security	Privilege	Interface Type
System Manager	32	l4_sys_free_clk	1/1/1	Secure	Privileged	OCP
FPGA Manager Data	32	l4_sys_free_clk	1/1/1	Secure	Privileged	OCP
FPGA Manager CSR	32	l4_sys_free_clk	1/1/1	Secure	Privileged	OCP
DAP	32	l4_sys_free_clk	1/1/1	Secure	Privileged	APB(15)
DMA Secure CSR	32	l4_main_clk	1/1/1	Boot Secure	User Mode	APB(15)
DMA Non-Secure CSR	32	l4_main_clk	1/1/1	Boot Secure	User Mode	APB(15)
SPI Slave 0/1	32	l4_main_clk	1/1/1	Boot Secure	User Mode	APB(15)
SPI Master 0/1	32	l4_main_clk	1/1/1	Boot Secure	User Mode	APB(15)
USB OTG CSR 0/1	32	l4_mp_clk	1/1/1	Boot Secure	User Mode	AHB
NAND CSR	32	l4_mp_clk	1/1/1	Boot Secure	User Mode	AHB
NAND Command and Data	32	l4_mp_clk	1/1/1	Boot Secure	User Mode	AHB
SD/MMC ECC	32	l4_mp_clk	1/1/2	Secure	Privileged	OCP
On Chip RAM ECC	32	l4_mp_clk	1/1/2	Secure	Privileged	OCP
DMA ECC	32	l4_mp_clk	1/1/2	Secure	Privileged	OCP
NAND ECC	32	l4_mp_clk	1/1/2	Secure	Privileged	OCP
USB OTG ECC	32	l4_mp_clk	1/1/2	Secure	Privileged	OCP
EMACS ECC	32	l4_mp_clk	1/1/2	Secure	Privileged	OCP
ACP	64	mpu_l2ram_clk	13/5/18	Secure	User Mode	AXI

(13) Acceptance is the maximum number of transactions accepted.

Slave	Interface Width	Clock	Acceptance (Read/ Write/Total) (13)	Security	Privilege	Interface Type
APB-DP	32	cs_pl4	1/1/1	Secure	Privileged	APB(15)
DDR	256	f2h_sdram_ clk[2:0]	16/16/16	Secure/Non-Secure	User Mode	Avalon
Lightweight HPS-to-FPGA Bridge	32	lwh2fpga_clk	8/8/8	Boot Secure	User Mode	AXI
HPS-to-FPGA Bridge	128/64/32	hps2fpga_clk	8/8/8	Boot Secure	User Mode	AXI
On-Chip RAM	64	l3_main_free_ clk	2/2/2	Secure/Non-Secure Per 4KB region	User Mode	AXI
STM	32	cs_at_clk	1/2/2	Secure/Non-Secure	User Mode	AXI

Note: APB-DP has no direct connection to the system interconnect.

System Interconnect Clocks

The system interconnect clocks are driven by the clock manager. The system interconnect's clocks are part of the NoC clock group, which is hardware-sequenced.

Related Information

Hardware Sequenced Clock Groups on page 2-11
For more information about the NoC clock group

List of Main L3 Interconnect Clocks

Table 7-7: Main L3 Interconnect Clocks

Clock Name	Description	Synchronous to l3_main_ free_clk
`l3_main_free_clk`	Clocks the main L3 interconnect	—
`l4_sys_free_clk`	Clocks the following interconnect components: • The L4 system bus • The interface to the DAP	Y

(13) Acceptance is the maximum number of transactions accepted.

Clock Name	Description	Synchronous to l3_main_free_clk
l4_main_clk	Clocks fast L4 peripherals on the L4 main bus Refer to "System Interconnect Master Properties" and "System Interconnect Slave Properties" for detailed peripheral-to-clock mappings.	Y
l4_mp_clk	Clocks the following interconnect components: • Mid-speed L4 peripherals on the L4 MP bus • The L4 AHB bus • The L4 ECC bus Refer to "System Interconnect Master Properties" and "System Interconnect Slave Properties" for detailed peripheral-to-clock mappings.	Y
l4_sp_clk	Clocks slow L4 peripherals on the L4 SP bus. Refer to "System Interconnect Master Properties" and "System Interconnect Slave Properties" for detailed peripheral-to-clock mappings.	Y
cs_at_clk	Coresight trace clock. Clocks the Coresight Embedded Trace Router (ETR) master interface.	Y
mpu_l2ram_clk	Clocks the MPU subsystem master interfaces.	N
fpga2hps_clk	Clocks the FPGA-to-HPS bridge.	N
hps2fpga_clk	Clocks the HPS-to-FPGA bridge.	N
lwh2fpga_clk	Clocks the lightweight HPS-to-FPGA bridge.	N

Related Information

- **System Interconnect Block Diagram** on page 7-2
- **System Interconnect Master Properties** on page 7-20
  Detailed peripheral-to-clock mappings
- **System Interconnect Slave Properties** on page 7-22
  Detailed peripheral-to-clock mappings

List of SDRAM L3 Interconnect Clocks

Table 7-8: SDRAM L3 Interconnect Clocks

Clock Name	Description	Synchronous to l3_main_free_clk
hmc_free_clk	Clock from the hard memory controller. Clocks the SDRAM L3 interconnect. The hmc_free_clk frequency is ½ the interface clock frequency.	N
f2s_sdram_clk [2:0]	Clocks the FPGA-to-SDRAM interfaces.	N

System Interconnect Resets

The system interconnect has one reset signal, l3_rst_n. The reset manager drives this signal to the system interconnect on a cold or warm reset.

Related Information

Reset Manager on page 3-1

Functional Description of the Rate Adapters

Rate adapters are used at points in the system interconnect where there are bandwidth discontinuities, to ensure efficient use of interconnect data pathways. They are placed where a low-bandwidth source feeds a high-bandwidth destination. For this reason, they are sometimes positioned at the response side (where the Master is faster) and sometimes at the request side (where the Slave is faster).

The following table shows the rate adapters.

Table 7-9: Rate Adapter Modules

Rate Adapter Module Name	Data Path
noc_mpu_m0_MPU_M1toDDRResp_main_RateAdapter	Data packets from the SDRAM L3 interconnect in response to the MPU
noc_mpu_m0_MPU_M0_rate_adResp_main_RateAdapter	Data packets from the L3 interconnect in response to the MPU
noc_mpu_m0_L4_MP_rate_ad_main_RateAdapter	Data packets carrying requests from L4 master peripherals to the L3 interconnect
noc_mpu_m0_fpga2soc_rate_ad_main_RateAdapter	Data packets carrying requests from the FPGA-to-HPS bridge master to the L3 interconnect
noc_mpu_m0_L3Tosoc2fpgaResp_main_RateAdapter	Data packets from the HPS-to-FPGA and lightweight HPS-to-FPGA bridges in response to the L3 interconnect
noc_mpu_m0_acp_rate_ad_main_RateAdapter	Data packets carrying requests from the L3 interconnect to the ACP

Related Information

Configuring the Rate Adapters on page 7-42

Functional Description of the Firewalls

Security

Slave Security

The system interconnect enforces security through the slave settings. The slave settings are controlled by the NoC Security Control Register (SCR) in the service network. Each L3 and L4 slave has its own security check and programmable security settings. After reset, every slave of the system interconnect is set to a secure state (referred to as boot secure). Only secure masters are allowed to access secure slaves.

The NoC implements five firewalls to check the security state of each slave, as listed in the following table. At reset time, all firewalls default to the secure state.

Table 7-10: NoC Firewalls

Name	Function
On-Chip RAM Firewall	Filter access to on-chip RAM
Peripherals Firewall	Filter access to slave peripherals (SPs) in the following buses: • L4 main bus • L4 master peripherals bus • L4 AHB bus • L4 slave peripherals bus
System Firewall	Filter access to system peripherals in the following components: • L4 system bus • L4 ECC bus • DAP
HPS-to-FPGA Firewall	Filter access to FPGA through the following bridges: • HPS-to-FPGA bridge • Lightweight HPS-to-FPGA bridge
DDR and DDR L3 Firewalls	Filter access to DDR SDRAM

At reset, the privilege filters are configured to allow certain L4 slaves to receive only secure transactions. Software must either configure bridges secure at startup, or reconfigure the privilege filters to accept nonsecure transactions. You can reconfigure the privilege filters through the `l4_priv` register in the `noc_l4_priv_l4_priv_filter` module.

To change the security state, you must perform a secure write to the appropriate SCR register of a secure slave. A nonsecure access to the SCR register of a secure slave triggers a response with random data. A nonsecure access does not trigger a bus error.

Related Information

l4_priv on page 7-191
Privilege bits, used to enable or disable nonsecure access to individual L4 slaves

Master Security

Masters of the system interconnect are either secure, nonsecure, or the security is set on a per transaction basis. L2 cache masters 0 and 1, the FPGA-to-HPS bridge, DMA, the Ethernet MACs, and the DAP perform secure and nonsecure accesses on a per-transaction basis. All other system interconnect masters perform nonsecure accesses.

Accesses to secure slaves by unsecure masters result in a response with random data.

Related Information

System Interconnect Master Properties on page 7-20

Functional Description of the SDRAM L3 Interconnect

The SDRAM L3 interconnect consists of two main blocks, serving the following two main functions:

- The SDRAM scheduler provides multi-port scheduling between the SDRAM L3 interconnect masters and the hard memory controller in the FPGA portion of the SoC. The SDRAM L3 interconnect is mastered by the MPU, the main L3 interconnect, and FPGA-to-SDRAM ports.
- The SDRAM adapter provides connectivity between the SDRAM L3 interconnect masters and the hard memory controller.

The SDRAM L3 interconnect also includes firewalls that can protect regions of SDRAM from unauthorized access.

The hard memory controller is physically located in the FPGA portion of the device, and therefore it is in a separate power domain from the HPS. The HPS cannot use the SDRAM L3 interconnect until the FPGA portion is powered up.

Functional Description of the SDRAM Scheduler

The SDRAM scheduler functions as a multi-port front end (MPFE), scheduling transactions from multiple masters to the SDRAM.

The SDRAM scheduler manages transactions to the memory access regions in the SDRAM. These memory regions are defined by the SDRAM L3 firewalls. The second-stage bootloader is expected to program the scheduler with the correct timings to implement optimal access patterns to the hard memory controller.

The SDRAM scheduler has the following features:

- Five input connections
  - One 64-bit connection from the MPU
  - One 64-bit connection from the main L3 interconnect
  - Two 128/64/32-bit connections from the FPGA
  - One 64/32-bit connection from the FPGA
- Single 256-bit connection to the SDRAM L3 adapter
  - Capable of issuing transactions at the memory device line rate
  - Traffic is comprised of aggregate inputs

Related Information

SDRAM L3 Firewalls on page 7-32

Monitors for Mutual Exclusion

The SDRAM scheduler implements support for mutually-exclusive (mutex) accesses on all ports to the SDRAM L3 interconnect.

The process for a mutually-exclusive access is as follows:

1. A master attempts to lock a memory location by performing an exclusive read from that address.
2. The master attempts to complete the exclusive operation by performing an exclusive write to the same address location.
3. The exclusive write access is signaled as:
   - Failed if another master has written to that location between the read and write accesses. In this case the address location is not updated.
   - Successful otherwise.

To support mutually-exclusive accesses from the MPU, the memory must be configured in the MMU page tables as normal memory, shareable, or non-cacheable.

Related Information

- **On-Chip Memory** on page 12-1
- **Embedded Peripheral IP User Guide**
  Contains detailed information about the Altera mutex core

Exclusive Access Support

To ensure mutually exclusive access to shared data, use the exclusive access support built into the SDRAM scheduler. The AXI buses that interface to the scheduler provide `ARLOCK[0]` and `AWLOCK[0]`, signals which are used by the scheduler to arbitrate for exclusive access to a memory location. The SDRAM scheduler contains six monitors. Each monitor can be used by any of the following exclusive-capable masters:

- CPU 0
- CPU 1
- FPGA-to-HPS bridge
- FPGA-to-SDRAM0 port
- FPGA-to-SDRAM1 port
- FPGA-to-SDRAM2 port

Each master can lock only one memory location at a time.

Quality of Service in the SDRAM Scheduler

You can apply QoS settings to each port of the SDRAM scheduler to control the bandwidth available to each master.

The SDRAM scheduler controls arbitration priorities internally for optimal end-to-end quality of service. Each master on the SDRAM scheduler has software-programmable QoS signals. These signals are propagated to the scheduler and used as arbitration criteria for access to SDRAM.

For information about programming quality of service for the FPGA-to-SDRAM masters, refer to "Functional Description of the QoS Generators".

Related Information

Functional Description of the QoS Generators on page 7-34

Functional Description of the SDRAM Adapter

The SDRAM adapter connects the SDRAM scheduler with the Hard Memory Controller.

The SDRAM adapter provides the following functionality:

- Connects the OCP master to the hard memory controller
- ECC generation, detection, and correction
- Operates at memory half rate
  - Matches interface frequency of the single port memory controller in the FPGA
  - Connectivity to the MPU, main L3 interconnect, and FPGA undergo clock crossing

ECC

The SDRAM Adapter ECC can detect and correct single-bit errors and detect double-bit errors.

The addresses are merged with data and are used for checking. This configuration detects if correct write data is written to an incorrect location in memory; and if correct data is read from an incorrect location in memory.

Note: Data and address errors are detected independently.

ECC Write Behavior

When data is written to SDRAM, the SDRAM controller generates an ECC based on the write data and the write address.

If the write transaction is a partial write (less than 64 bits wide), the SDRAM adapter implements it as a read-modify-write (RMW) sequence, as follows:

- Reads existing data from the specified address
- Combine the write data with the existing data
- Generates the ECC based on the combined data and the write address
- Writes the combined data back to the write address

ECC Read Behavior

When data is read from SDRAM, the SDRAM controller checks the ECC to determine if the data or address is incorrect. It handles the following cases:

- If the SDRAM controller finds a single-bit data error, it corrects in the data returned to the master. You can also enable the SDRAM adapter to write the corrected data back to memory.
- If the SDRAM controller finds a double-bit data error, the SDRAM L3 interconnect issues an interrupt. Double-bit errors cannot be corrected.
- If the SDRAM controller finds an error in the address, indicating an address bit upset between the adapter and the HMC, the SDRAM L3 interconnect hardware issues a bus error.

SDRAM Adapter Interrupt Support

The SDRAM adapter supports the following three interrupts:

- The status interrupt occurs when:
  - Calibration is complete.
  - The ECC is unable to schedule an auto-correction write-back to memory. This occurs only when the auto-write-back FIFO is full.
- The ECC read-back interrupt occurs when an ECC single-bit error is detected in the read data. When this happens, the return data is corrected and returned to the NoC.
- The double-bit or fatal error interrupt occurs when any of the following three errors happens:
  - A double-bit error in the read data has been detected, which cannot be corrected.
  - A single-bit error has been detected in the address field. This means that the data that the adapter is returning has no bit errors, but is not the requested data. When this happens, the adapter returns a data error along with the data.
  - Any of the DDR4 devices have triggered their ALERT pins.
    - Address or command parity check has failed
    - Write data CRC check has failed
    - Cannot gracefully recover because SDRAMs are not providing feedback on failure case

SDRAM Adapter Clocks

All the logic in the SDRAM adapter is synchronous and effectively running off a single clock, `hmc_free_clk`, which is provided by the hard memory controller.

SDRAM L3 Firewalls

All data that is routed to the SDRAM scheduler must pass through the firewalls.

The SDRAM L3 firewalls define memory access regions in the SDRAM. Each SDRAM L3 interconnect master has its own memory access regions, independent of the other masters. The firewalls define whether each memory access region is protected or unprotected relative to its master. The number of available memory access regions for each master is shown in the following table.

Table 7-11: Memory Access Regions for SDRAM Masters

SDRAM L3 Interconnect Master	Number of Memory Access Regions
MPU	4
Main L3 interconnect (including the FPGA-to-HPS bridge)	8
FPGA-to-SDRAM port 0	4
FPGA-to-SDRAM port 1	4
FPGA-to-SDRAM port 2	4

The SDRAM L3 interconnect regulates access to the hard memory controller with the firewalls, which support secure regions in the SDRAM address space. Accesses to the SDRAM pass through the firewalls and then through the scheduler.

Related Information

SoC Security on page 6-1

SDRAM L3 Interconnect Resets

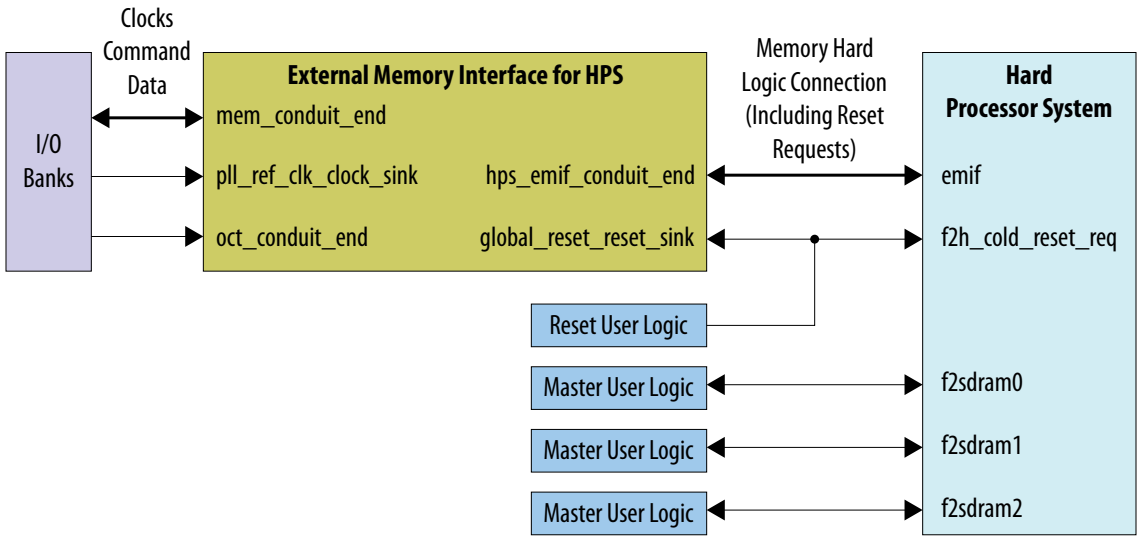
The reset signal `l3_rst_n` resets the system interconnect and the SDRAM L3 interconnect, but not the hard memory controller.

The reset signal in the hard memory controller is automatically connected to the SDRAM L3 interconnect when you instantiate the HPS component. For information about resetting the hard memory controller, refer to the *External Memory Interfaces in Arria 10 Devices* chapter of the *Arria 10 Core Fabric and General Purpose I/O Handbook*.

Soft logic in the FPGA must support the `global_reset_n` signal correctly. Refer to the *Instantiating the HPS Component* chapter for information about `global_reset_n`.

To optionally preserve the contents of the SDRAM on reset, refer to "Reset Handshaking" in the *Reset Manager* chapter.

Figure 7-11: Recommended SDRAM Reset Connections



Note: It is important to connect the reset user logic directly to both the HPS and the hard memory controller. If the hard memory controller is reset while the HPS is still running, the HPS is unable to access any external SDRAM memory.

Related Information

- **System Interconnect Resets** on page 7-26
- **Instantiating the HPS Component** on page 27-1
  Details about hard memory controller reset connections and correct support for global_reset_n
- **Reset Handshaking** on page 3-21
- **Arria 10 Core Fabric and General Purpose I/O Handbook**
  Refer to the *External Memory Interfaces in Arria 10 Devices* chapter for information about resetting the hard memory controller.

Functional Description of the QoS Generators

Quality of service is managed by information generated at each initiator interface by a QoS generator.

At the entry points to the system interconnect, the QoS generators create signals and other information that control how much interconnect bandwidth is allocated to transactions. The QoS information is used at each interconnect arbitration node to assign a packet priority.

Related Information

- **About Quality of Service and Arbitration** on page 7-11
- **Quality of Service in the SDRAM Scheduler** on page 7-30

QoS Generators in the Interconnect Architecture

The principal masters into the system interconnect are equipped with QoS generators.

The following table shows the network initiators (masters) that support QoS generation.

Table 7-12: QoS Generators on Initiators

Master Name(s)	Default QoS Generator Mode	Clock
MPU Subsystem L2 cache M0/1	Limiter	`mpu_l2ram_clk`
FPGA-to-HPS Bridge	Limiter	`fpga2hps_clk`
FPGA-to-SDRAM Ports	Regulator	`f2h_sdram_clk`
DMA	Limiter	`l4_main_clk`
EMAC 0/1/2	Regulator	`l4_mp_clk`
USB OTG 0/1	Regulator	`l4_mp_clk`
NAND	Regulator	`l4_mp_clk`
SD/MMC	Regulator	`l4_mp_clk`

Related Information

QoS Generator Modes on page 7-35

QoS Mechanisms

The QoS generators influence arbitration priorities by manipulating urgency, pressure, and hurry.

When two or more packets compete for an interconnect resource at an arbitration node, the arbitration logic selects the packet with the highest priority. QoS generators can control the behavior of the arbitration logic by modifying the priority of packets through urgency, pressure, and hurry.

The QoS controls for each initiator are separated into read and write QoS controls.

Related Information

Functional Description of the Arbitration Logic on page 7-39

Packet Urgency

A QoS generator assigns an urgency level to each packet when the packet data arrives at an interconnect master port.

The urgency level is attached to the packet as long as it is moving through the interconnect. Urgency is the primary QoS mechanism.

Pressure Propagation

Each active channel in the interconnect has a pressure level, which can change depending on the traffic through the channel.

The pressure is based on the urgency of packets waiting for that channel. If a higher-priority packet requests a channel that is already active, the channel's urgency increases.

When a packet arrives at an arbitration node, its priority is either its urgency or its pressure, whichever is greater. If the pressure is greater than the urgency, the arbitration logic moves the packet through more quickly to free up the channel for the higher-urgency packet that created the pressure.

Hurry

Each initiator has a hurry level, which can change depending on the bandwidth usage and the QoS generator mode.

When the hurry level is increased at an initiator, it has the effect of setting the urgency of that initiator's transactions to at least the same level as the hurry. As a result, the targets respond to the transactions more quickly.

QoS Generator Modes

Each QoS generator can operate in any of the following modes:

- Limiter
- Regulator
- Fixed

For the default QoS generator mode for each initiator, refer to "QoS Generators in the Interconnect Architecture".

QoS generators can also be set to bypass mode. In this case, QoS signals are provided by the component connected to the master interface.

Related Information

QoS Generators in the Interconnect Architecture on page 7-34
Default QoS modes for each initiator

Limiter Mode

In limiter mode, the QoS generator controls the initiator's access to the interconnect, by monitoring the amount of data transmitted and received in a sliding time window.

The purpose of limiter mode is to place a hard bound on the initiator's throughput. This avoids flooding the interconnect with transactions from that initiator. When a programmable bandwidth threshold is exceeded, the initiator stops accepting new requests. As the time window moves forward, the total bandwidth usage in the window falls, and the initiator resumes accepting requests.

In limiter mode, the QoS generator combines read and write requests to determine the bandwidth usage.

Related Information

Programming QoS Limiter Mode on page 7-36

Regulator Mode

In regulator mode, the QoS generator maintains bandwidth usage to transaction targets at a programmed level, by adjusting urgency, pressure, and hurry.

Regulator mode controls bandwidth usage more smoothly than limiter mode. This mode is useful in achieving real-time performance.

When the initiator's bandwidth usage exceeds the specified threshold, the QoS decreases the urgency, pressure, and hurry levels. When bandwidth usage is too low, the QoS increases urgency, pressure and hurry. Software can program the desired bandwidth threshold, the saturation, and maximum and minimum levels for urgency, pressure, and hurry.

Related Information

Programming QoS Regulator Mode on page 7-37

Fixed Mode

In fixed mode, the QoS generator assigns a fixed urgency to each packet. Software can configure separate urgencies for read and write transactions.

Related Information

Programming QoS Fixed Mode on page 7-38

Controlling Quality of Service from Software

You can programmatically configure the QoS generator for each initiator through the QoS registers.

Note: Before accessing the registers for any QoS generator, you must ensure that the corresponding peripheral is not in reset. Otherwise, the register access results in a bus error, causing a CPU fault.

Programming QoS Limiter Mode

To put the QoS generator in limiter mode, set the registers as shown in the following table.

Table 7-13: QoS Generator Register Values for Limiter Mode

Register	Field	Value
`I_main_QosGenerator_Mode`	`MODE`	1
`I_main_QosGenerator_Priority`	`P0`	Urgency for write transations
`I_main_QosGenerator_Priority`	`P1`	Urgency for read transations
`I_main_QosGenerator_Bandwidth`	`BANDWIDTH`	Maximum bandwidth, in units of (bytes/cycle)/256

Register	Field	Value
I_main_QosGenerator_Saturation	SATURATION	Measurement window for bandwidth, in units of bytes/16. This register specifies the maximum number of bytes that the initiator can transmit or receive at full speed before the limiter triggers.

When you switch QoS modes, the bandwidth counter is reset.

Related Information

- **QoS Generators in the Interconnect Architecture** on page 7-34
  List of clock domains for each initiator
- **QoS Generator Registers** on page 7-38
  List of QoS registers

Programming QoS Regulator Mode

To put the QoS generator in regulator mode, set the registers as shown in the following table.

Table 7-14: QoS Generator Register Values for Regulator Mode

Register	Field	Value
I_main_QosGenerator_Mode	MODE	3
I_main_QosGenerator_Priority	P0	Packet urgency when the actual throughput exceeds the threshold set in I_main_QosGenerator_Bandwidth.
I_main_QosGenerator_Priority	P1	Packet urgency when the actual throughput is less than the threshold set in I_main_QosGenerator_Bandwidth. P0 must be less than or equal to P1.
I_main_QosGenerator_Bandwidth	BANDWIDTH	Desired throughput, in units of (bytes/cycle)/256
I_main_QosGenerator_Saturation	SATURATION	Measurement window for bandwidth, in units of bytes/16.

When you switch QoS modes, the bandwidth counter is reset.

Related Information

- **QoS Generators in the Interconnect Architecture** on page 7-34
  List of clock domains for each initiator
- **QoS Generator Registers** on page 7-38
  List of QoS registers

Programming QoS Fixed Mode

To put the QoS generator in fixed mode, set the registers as shown in the following table.

Table 7-15: QoS Generator Register Values for Fixed Mode

Register	Field	Value
I_main_QosGenerator_Mode	MODE	0
I_main_QosGenerator_Priority	P0	Urgency for write transations
I_main_QosGenerator_Priority	P1	Urgency for read transations

Related Information

QoS Generator Registers on page 7-38
List of QoS registers

QoS Generator Registers

There is a register set for each QoS generator.

Each QoS generator's register set includes the register types shown in the following table.

Table 7-16: QoS Generator Register Types

Name	Purpose
I_main_QosGenerator_Priority	Specifies packet urgencies. Register usage depends on the QoS generator mode.
I_main_QosGenerator_Mode	Specifies the QoS generator mode (limiter, regulator, fixed, or bypass).
I_main_QosGenerator_Bandwidth	Specifies the bandwidth threshold for limiter and regulator modes.
I_main_QosGenerator_Saturation	Specifies the bandwidth measurement time window for limiter and regulator modes.

The actual register names consist of the register type plus a prefix specific to the QoS generator. For example, the registers for MPU subsystem L2 cache master 0 are:

- mpu_m0_I_main_QosGenerator_Priority
- mpu_m0_I_main_QosGenerator_Mode
- mpu_m0_I_main_QosGenerator_Bandwidth
- mpu_m0_I_main_QosGenerator_Saturation
- mpu_m0_I_main_QosGenerator_ExtControl

The following table lists the prefixes for all QoS generators.

Table 7-17: QoS Register Name Prefixes

Initiator	Register Name Prefix
MPU Subsystem L2 Cache Master 0	`mpu_m0_`
MPU Subsystem L2 Cache Master 1	`mpu_m1_`
FPGA-to-HPS Bridge	`fpga2soc_axi32_`
FPGA-to-HPS Bridge	`fpga2soc_axi64_`
FPGA-to-HPS Bridge	`fpga2soc_axi128_`
DMA Controller	`dma_m0_`
Ethernet MAC 0	`emac0_m_`
Ethernet MAC 1	`emac1_m_`
Ethernet MAC 2	`emac2_m_`
USB-2 OTG 0	`usb0_m_`
USB-2 OTG 1	`usb1_m_`
NAND Flash Controller	`nand_m_`
SD/MMC Controller	`sdmmc_m_`
FPGA-to-SDRAM Port 0	`fpga2sdram0_axi32_`
FPGA-to-SDRAM Port 0	`fpga2sdram0_axi64_`
FPGA-to-SDRAM Port 0	`fpga2sdram0_axi128_`
FPGA-to-SDRAM Port 1	`fpga2sdram1_axi32_`
FPGA-to-SDRAM Port 1	`fpga2sdram1_axi64_`
FPGA-to-SDRAM Port 2	`fpga2sdram2_axi32_`
FPGA-to-SDRAM Port 2	`fpga2sdram2_axi64_`
FPGA-to-SDRAM Port 2	`fpga2sdram2_axi128_`

QoS Address Map and Register Descriptions

Functional Description of the Arbitration Logic

The interconnect arbitration logic handles situations in which multiple simultaneous packets need the same interconnect resource.

The system interconnect contains arbitration nodes at each point where multiple packets might demand the same resource. Each arriving packet has a priority. When multiple packets of different priorities arrive simultaneously at a node, the arbitration logic grants the resource to the highest-priority packet.

If there are simultaneous packets with the same priority, the system interconnect uses a simple round-robin (least recently used) algorithm to select the packet.

Each packet's priority is determined by urgency, pressure, and hurry, as described in "QoS Mechanisms".

Related Information

QoS Mechanisms on page 7-34

Functional Description of the Observation Network

Probes

The system interconnect includes the probes shown in the following table.

Table 7-18: Observation Probes

Probe Name	Probe ID	Probe Type	Probe Target(s)	Packet Tracing and Statistics	Transaction Profiling	Error Logging
MPU	6	Packet	MPU master 0	•		
SOC2FPGA	5	Packet	• HPS-to-FPGA bridge • Lightweight HPS-to-FPGA bridge	•		
DDR	10	Packet	SDRAM adapter	•		
EMAC	4	Packet	EMAC0		•	
EMAC	4	Packet	• EMAC1 • EMAC2	•		
ERROR	n/a	Error	• HPS-to-FPGA bridge • Lightweight HPS-to-FPGA bridge`			•

The probe ID is available on the ATB trace bus.

Packet Tracing, Profiling, and Statistics

Packet probes perform packet tracing, profiling and statistics. The following table shows how each packet probe is configured. All statistics counters are 16 bits wide, and all probes can generate level alarms from the statistics counter. All probes use the CoreSight cross-trigger protocol.

Table 7-19: Packet Probe Characteristics

Probe Name	Purpose	Number of Filters	Statistics Counters Can Count Enabled Bytes	Packet Payloads Included in Trace Data	Number of Statistics Counters
MPU	MPU	1	No	No	4
SOC2FPGA	HPS-to-FPGA bridge	2	No	No	4

Probe Name	Purpose	Number of Filters	Statistics Counters Can Count Enabled Bytes	Packet Payloads Included in Trace Data	Number of Statistics Counters
EMAC	Ethernet MAC	2	No	No	4
DDR	SDRAM	4	Yes	Yes	2

Packet Filtering

You can set up filters to control how the observation network handles traced packets.

Filters can perform the following tasks:

- Select which packets the observation network routes to CoreSight
- Trigger a trace alarm when a packet meets specified criteria

Statistics Collection

To collect packet statistics, you specify packet criteria and set up counters for packets that meet those criteria. You can set up the observation network to trigger an alarm when a counter reaches a specified level.

Transaction Profiling

A transaction probe is available on Ethernet MAC 0. You can use the transaction probe to measure either the transaction latency or the number of pending packets on EMAC0. Data are collected as a histogram.

The EMAC0 transaction probe is configured as shown in the following table.

Table 7-20: EMAC0 Transaction Probe Configuration

Number of simultaneously tracked transactions	4
Width of counters	10 bits
Available delay threshholds	64, 128, 256, 512
Available pending transaction count thresholds	2, 4, 8
Number of comparators	3
Number of histogram bins	4

Profiling Transaction Latency

In latency mode (also called delay mode), one of the four delay threshold values can be chosen for each comparator. The threshold values represent the number of clock cycles that a transaction takes from the time the request is issued to the time the response is returned.

Profiling Pending Transactions

In pending transaction mode, one of the three available transaction count threshold values can be chosen for each comparator. The threshold values represent the number of requests pending on EMAC0.

Packet Alarms

Packet alarms can be used to trigger software interrupts.

The following types of alarms are available:

- Trace alarms—You can examine trace alarms through the system interconnect registers.
- Statistics alarms

Error Logging

The error probe logs errors on initiators, targets, and firewalls.

Configuring the System Interconnect

Related Information

SoC Security on page 6-1

Configuring the Rate Adapters

The default settings for the rate adapters are as shown in the following table.

Table 7-21: Rate Register Default Settings

Rate Register Name	Default
MPU_M1toDDRResp_main_RateAdapter_Rate	0x0
MPU_M0_rate_adResp_main_RateAdapter_Rate	0x0
L4_MP_rate_ad_main_RateAdapter_Rate	0x100
fpga2soc_rate_ad_main_RateAdapter_Rate	0x0
L3Tosoc2fpgaResp_main_RateAdapter_Rate	0x0
acp_rate_ad_main_RateAdapter_Rate	0x0

Configuring the SDRAM Scheduler

FPGA Port Configuration

The FPGA has three outputs that pass through the firewall before connecting to the SDRAM scheduler.

You can configure the FPGA-to-SDRAM (F2SDRAM) ports to data widths of 32, 64, or 128 bits:

- F2SDRAM 0 - 32-, 64-, or 128-bit data widths
- F2SDRAM 1 - 32- or 64-bit data widths
- F2SDRAM 2 - 32-, 64-, or 128-bit data widths

There are four port configurations that are a combination of the SDRAM ports 0 - 2 that you are able to select. Once a port configuration is selected, you can choose to disable ports that you do not need.

Note: The total data width of all interfaces is limited to a maximum of 256 bits in the read direction and 256 bits in the write direction.

Port Configuration	F2SDRAM 0	F2SDRAM 1	F2SDRAM 2
1	32-bit	32-bit	32-bit
2	64-bit	64-bit	64-bit
3	128-bit	unused	128-bit
4	128-bit	32-bit	64-bit

Memory Timing Configuration

The SDRAM L3 interconnect provides fully-programmable timing parameter support for all JEDEC-specified timing parameters.

The following lists the handoff information used to control the SDRAM scheduler:

- The scheduler is aware of the SDRAM timing so that it can guide traffic into the hard memory controller.
- This information is not used to control the subsystem that sets up memory timings in the hard memory controller hardware.

Configuring SDRAM Burst Sizes

You can optimize memory throughput by adjusting the SDRAM burst lengths for read and write transactions. The best burst length values are application-dependent. Altera recommends that you follow guidelines in *Arria 10 SoC Device Design Guidelines* to determine the optimal burst lengths.

Related Information

- **Arria 10 SoC Device Design Guidelines**
- **HPS-to-FPGA Bridges Design Example**
  Download **Arria 10 FPGA-to-HPS Bridges design example (.zip file)** for use with the *Arria 10 SoC Device Design Guidelines*.

Configuring the Hard Memory Controller

SDRAM Adapter Memory Mapped Registers

The SDRAM adapter memory mapped registers (MMRs) are used for configuring and reading ECC status; and configuring data width adaptation for 16-, 32-, and 64-bit data widths.

Hard Memory Controller Memory Mapped Registers

The FPGA hard memory controller MMRs are used for determining the state of the hard memory controller interface, and for triggering the hard memory controller into a reset state along with the I/O. The FPGA hard memory controller MMRs can be programmed as secure or non-secure.

Related Information

Reset Handshaking on page 3-21

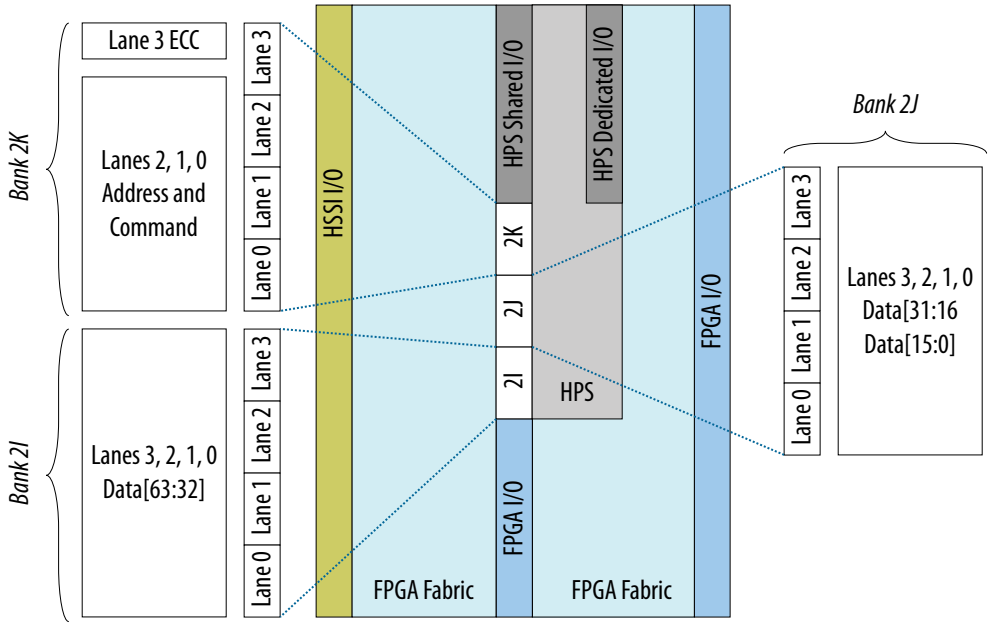
Sharing I/Os between the EMIF and the FPGA

Arria 10 SoC devices have three modular I/O banks to connect the HPS to an SDRAM (2K, 2J, and 2I) through a dedicated HPS EMIF.

Each bank has four I/O lanes that correspond to:

- Lane 3: IO[47:36]
- Lane 2: IO[35:24]
- Lane 1: IO[23:12]
- Lane 0: IO[11:0]

Figure 7-12: HPS Hard Memory Controller



To use SDRAM, you instantiate the HPS EMIF in Qsys, by selecting the **Arria 10 External Memory Interfaces for HPS** component. Quartus Prime software assigns the correct banks and lanes for the SDRAM I/O.

I/Os in these three banks can be shared between the HPS EMIF and FPGA. Quartus Prime software enforces the following guidelines on shared I/Os:

- Modular I/O banks 2K, 2J and 2I can be used entirely as FPGA GPIO when there is no HPS EMIF in the system.
- Bank 2K:
  - If there is an HPS EMIF in the system, lane 3 is used for ECC for the SDRAM. Unused pins in lane 3 may be used as FPGA inputs only.
  - Lanes 2, 1, and 0 are used for address and command for the SDRAM. Unused pins in these lanes may be used as FPGA inputs or outputs.
- Bank 2J:
  - If there is an HPS EMIF in the system, bank 2J is used for data bits [31:0].
    - With 16-bit data widths, unused pins in the two lanes of bank 2J used for data can be used as inputs only. The pins in the remaining two lanes can be used as FPGA inputs or outputs.
    - With 32-bit data widths, unused pins bank 2J can be used as FPGA inputs only.
- Bank 2I:
  - If there is an HPS EMIF in the system, bank 2I is used for data bits [63:32]
    - With 64-bit data widths, unused pins in bank 2I can be used as FPGA inputs only.
    - With 32- or 16-bit data widths, bank 2I can be used as FPGA inputs or outputs.

System Interconnect Address Map and Register Definitions

For complete HPS address map and register definitions, refer to the **Arria 10 HPS Address Map and Register Definitions** .

io48_hmc_mmr Address Map

Module Instance	Base Address	End Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFAFFF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
dbgcfg0 on page 7-63	0x0	32	RW	0x0	
dbgcfg1 on page 7-63	0x4	32	RW	0x0	
dbgcfg2 on page 7-64	0x8	32	RW	0x0	
dbgcfg3 on page 7-65	0xC	32	RW	0x0	
dbgcfg4 on page 7-65	0x10	32	RW	0x0	

Register	Offset	Width	Access	Reset Value	Description
dbgcfg5 on page 7-66	0x14	32	RW	0x0	
dbgcfg6 on page 7-67	0x18	32	RW	0x0	
reserve0 on page 7-67	0x1C	32	RW	0x0	
reserve1 on page 7-68	0x20	32	RW	0x0	
reserve2 on page 7-69	0x24	32	RW	0x0	
ctrlcfg0 on page 7-69	0x28	32	RW	0x0	
ctrlcfg1 on page 7-71	0x2C	32	RW	0x0	
ctrlcfg2 on page 7-73	0x30	32	RW	0x0	
ctrlcfg3 on page 7-75	0x34	32	RW	0x0	
ctrlcfg4 on page 7-77	0x38	32	RW	0x0	
ctrlcfg5 on page 7-79	0x3C	32	RW	0x0	
ctrlcfg6 on page 7-80	0x40	32	RW	0x0	
ctrlcfg7 on page 7-81	0x44	32	RW	0x0	
ctrlcfg8 on page 7-82	0x48	32	RW	0x0	
ctrlcfg9 on page 7-83	0x4C	32	RW	0x0	
dramtiming0 on page 7-84	0x50	32	RW	0x0	
dramodt0 on page 7-85	0x54	32	RW	0x0	
dramodt1 on page 7-86	0x58	32	RW	0x0	
sbcfg0 on page 7-87	0x5C	32	RW	0x0	
sbcfg1 on page 7-88	0x60	32	RW	0x0	

Register	Offset	Width	Access	Reset Value	Description
sbcfg2 on page 7-89	0x64	32	RW	0x0	
sbcfg3 on page 7-90	0x68	32	RW	0x0	
sbcfg4 on page 7-90	0x6C	32	RW	0x0	
sbcfg5 on page 7-91	0x70	32	RW	0x0	
sbcfg6 on page 7-92	0x74	32	RW	0x0	
sbcfg7 on page 7-93	0x78	32	RW	0x0	
caltiming0 on page 7-94	0x7C	32	RW	0x0	
caltiming1 on page 7-95	0x80	32	RW	0x0	
caltiming2 on page 7-96	0x84	32	RW	0x0	
caltiming3 on page 7-97	0x88	32	RW	0x0	
caltiming4 on page 7-98	0x8C	32	RW	0x0	
caltiming5 on page 7-98	0x90	32	RW	0x0	
caltiming6 on page 7-99	0x94	32	RW	0x0	
caltiming7 on page 7-100	0x98	32	RW	0x0	
caltiming8 on page 7-101	0x9C	32	RW	0x0	
caltiming9 on page 7-102	0xA0	32	RW	0x0	
caltiming10 on page 7-103	0xA4	32	RW	0x0	
dramaddrw on page 7-104	0xA8	32	RW	0x0	
sideband0 on page 7-105	0xAC	32	RW	0x0	
sideband1 on page 7-106	0xB0	32	RW	0x0	
sideband2 on page 7-106	0xB4	32	RW	0x0	

Register	Offset	Width	Access	Reset Value	Description
sideband3 on page 7-107	0xB8	32	RW	0x0	
sideband4 on page 7-108	0xBC	32	RW	0x0	
sideband5 on page 7-108	0xC0	32	RW	0x0	
sideband6 on page 7-109	0xC4	32	RO	0x0	
sideband7 on page 7-110	0xC8	32	RO	0x0	
sideband8 on page 7-111	0xCC	32	RO	0x0	
sideband9 on page 7-111	0xD0	32	RO	0x0	
sideband10 on page 7-112	0xD4	32	RO	0x0	
sideband11 on page 7-113	0xD8	32	RO	0x0	
sideband12 on page 7-114	0xDC	32	RW	0x0	
sideband13 on page 7-114	0xE0	32	RW	0x0	
sideband14 on page 7-115	0xE4	32	RW	0x0	
sideband15 on page 7-116	0xE8	32	RW	0x0	
dramsts on page 7-117	0xEC	32	RO	0x0	
dbgdone on page 7-118	0xF0	32	RO	0x0	
dbgsignals on page 7-119	0xF4	32	RO	0x0	
dbgreset on page 7-119	0xF8	32	RW	0x0	
dbgmatch on page 7-120	0xFC	32	RO	0x0	
counter0mask on page 7-121	0x100	32	RW	0x0	

Register	Offset	Width	Access	Reset Value	Description
counter1mask on page 7-122	0x104	32	RW	0x0	
counter0match on page 7-122	0x108	32	RW	0x0	
counter1match on page 7-123	0x10C	32	RW	0x0	
niosreserve0 on page 7-124	0x110	32	RW	0x0	
niosreserve1 on page 7-124	0x114	32	RW	0x0	
niosreserve2 on page 7-125	0x118	32	RW	0x0	

io48_hmc_mmr Summary

Base Address: 0xFFCFA000

Register Address Offset	Bit Fields															
i_io48_hmc_mmr_io48_mmr																
dbgcfg0 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							cfg_dbg_mode 0x0				cfg_cmd_driver_sel 0x0	cfg_loopback_en 0x0	cfg_mmr_driver_sel 0x0	cfg_prbs_ctrl_sel 0x0	cfg_wdata_driver_sel 0x0
dbgcfg1 0x4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cfg_dbg_ctrl 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cfg_dbg_ctrl 0x0															

Register Address Offset	Bit Fields															
dbgcfg2 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cfg_bist_cmd0_u															
	0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cfg_bist_cmd0_u															
	0x0															
dbgcfg3 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cfg_bist_cmd0_l															
	0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cfg_bist_cmd0_l															
	0x0															
dbgcfg4 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cfg_bist_cmd1_u															
	0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cfg_bist_cmd1_u															
	0x0															
dbgcfg5 0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cfg_bist_cmd1_l															
	0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cfg_bist_cmd1_l															
	0x0															
dbgcfg6 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cfg_dbg_out_sel															
	0x0															
reserve0 0x1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cfg_reserve0															
	0x0															

Register Address Offset	Bit Fields															
reserve1 0x20	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cfg_reserve1 0x0															
reserve2 0x24	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cfg_reserve2 0x0															
ctrlcfg0 0x28	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved			cfg_dbc2_burst_length 0x0					cfg_dbc1_burst_length 0x0					cfg_dbc0_burst_length 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cfg_dbc0_burst_length 0x0		cfg_ctrl_burst_length 0x0					cfg_ac_pos 0x0		cfg_dimm_type 0x0			cfg_mem_type 0x0			
ctrlcfg1 0x2C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	cfg_dbc3_enable_dm 0x0	cfg_dbc2_enable_dm 0x0	cfg_dbc1_enable_dm 0x0	cfg_dbc0_enable_dm 0x0	cfg_ctrl_enable_dm 0x0	cfg_dqstrk_en 0x0	cfg_starve_limit 0x0						cfg_reorder_read 0x0	cfg_dbc3_reorder_rdata 0x0	cfg_dbc2_reorder_rdata 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cfg_dbc1_reorder_rdata 0x0	cfg_dbc0_reorder_rdata 0x0	cfg_ctrl_reorder_rdata 0x0	cfg_reorder_data 0x0	cfg_dbc3_enable_ecc 0x0	cfg_dbc2_enable_ecc 0x0	cfg_dbc1_enable_ecc 0x0	cfg_dbc0_enable_ecc 0x0	cfg_ctrl_enable_ecc 0x0	cfg_addr_order 0x0		cfg_dbc3_burst_length 0x0				

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved					cfg_dbc3_pipe_lat 0x0			cfg_dbc2_pipe_lat 0x0			cfg_dbc1_pipe_lat 0x0			cfg_dbc0_pipe_lat 0x0	
ctrlcfg2 0x30	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cfg_dbc0_pipe_lat 0x0	cfg_dbc2ctrl_sel 0x0		cfg_dbc3_ctrl_sel 0x0	cfg_dbc2_ctrl_sel 0x0	cfg_dbc1_ctrl_sel 0x0	cfg_dbc0_ctrl_sel 0x0	cfg_ctrl2dbc_switch1 0x0		cfg_ctrl2dbc_switch0 0x0		cfg_dbc3_output_regd 0x0	cfg_dbc2_output_regd 0x0	cfg_dbc1_output_regd 0x0	cfg_dbc0_output_regd 0x0	cfg_ctrl_output_regd 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	cfg_rld3_multibank_mode 0x0			cfg_geardn_en 0x0	cfg_open_page_en 0x0	cfg_arbiter_type 0x0	cfg_dbc3_dualport_en 0x0	cfg_dbc2_dualport_en 0x0	cfg_dbc1_dualport_en 0x0	cfg_dbc0_dualport_en 0x0	cfg_ctrl_dualport_en 0x0	cfg_dbc3_in_protocol 0x0	cfg_dbc2_in_protocol 0x0	cfg_dbc1_in_protocol 0x0	cfg_dbc0_in_protocol 0x0
ctrlcfg3 0x34	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cfg_ctrl_in_protocol 0x0	cfg_dbc3_cmd_rate 0x0			cfg_dbc2_cmd_rate 0x0			cfg_dbc1_cmd_rate 0x0			cfg_dbc0_cmd_rate 0x0			cfg_ctrl_cmd_rate 0x0		
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cfg_dbc3_slot_offset 0x0		cfg_dbc2_slot_offset 0x0		cfg_dbc1_slot_offset 0x0		cfg_dbc0_slot_offset 0x0		cfg_ctrl_slot_offset 0x0		cfg_dbc3_slot_rotate_en 0x0			cfg_dbc2_slot_rotate_en 0x0		
ctrlcfg4 0x38	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cfg_dbc1_slot_rotate_en 0x0			cfg_dbc0_slot_rotate_en 0x0			cfg_ctrl_slot_rotate_en 0x0			cfg_pingpong_mode 0x0		cfg_tile_id 0x0				
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
ctrlcfg5 0x3C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved			cfg_dbc3_rc_en 0x0	cfg_dbc2_rc_en 0x0	cfg_dbc1_rc_en 0x0	cfg_dbc0_rc_en 0x0	cfg_ctrl_rc_en 0x0	cfg_row_cmd_slot 0x0			cfg_col_cmd_slot 0x0				

Register Address Offset	Bit Fields															
ctrlcfg6 0x40	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cfg_cs_chip 0x0															
ctrlcfg7 0x44	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	cfg_wb_reserved_entry 0x0							cfg_rb_reserved_entry 0x0							cfg_clkgating_en 0x0
ctrlcfg8 0x48	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													cfg_addr_mplx_en 0x0	cfg_ck_inv 0x0	cfg_3ds_en 0x0
ctrlcfg9 0x4C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															cfg_dfx_bypass_en 0x0
dramtiming0 0x50	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved													cfg_mem_clk_disable_entry_cycles 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cfg_mem_clk_disable_entry_cycles 0x0			cfg_power_saving_exit_cycles 0x0						cfg_tcl 0x0						

Register Address Offset	Bit Fields															
dramodt0 0x54	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cfg_read_odt_chip 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cfg_write_odt_chip 0x0															
dramodt1 0x58	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved								cfg_rd_odt_period 0x0						cfg_wr_odt_period 0x0	
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cfg_wr_odt_period 0x0				cfg_rd_odt_on 0x0						cfg_wr_odt_on 0x0					
sbcfg0 0x5C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cfg_rld3_refresh_seq1 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cfg_rld3_refresh_seq0 0x0															
sbcfg1 0x60	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cfg_rld3_refresh_seq3 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cfg_rld3_refresh_seq2 0x0															
sbcfg2 0x64	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								cfg_srf_entry_exit_block 0x0		cfg_srf_auto exit_en 0x0	cfg_user_rfsh_en 0x0	cfg_sb_cg_disable 0x0	cfg_mps_dqstrk_disable 0x0	cfg_mps_zqcal_disable 0x0	cfg_srf_zqcal_disable 0x0

Register Address Offset	Bit Fields															
sbcfg3 0x68	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved												cfg_sb_ddr4_mr3 0x0			
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cfg_sb_ddr4_mr3 0x0															
sbcfg4 0x6C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved												cfg_sb_ddr4_mr4 0x0			
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cfg_sb_ddr4_mr4 0x0															
sbcfg5 0x70	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														cfg_period_dqstrk_ctrl_en 0x0	cfg_short_dqstrk_ctrl_en 0x0
sbcfg6 0x74	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cfg_t_param_dqstrk_to_valid 0x0								cfg_t_param_dqstrk_to_valid_last 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cfg_period_dqstrk_interval 0x0															
sbcfg7 0x78	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved									cfg_rfsh_warn_threshold 0x0						

Register Address Offset	Bit Fields															
caltiming0 0x7C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved		cfg_t_param_act_to_act_diff_bg 0x0						cfg_t_param_act_to_act_diff_bank 0x0						cfg_t_param_act_to_act 0x0	
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cfg_t_param_act_to_act 0x0				cfg_t_param_act_to_pch 0x0						cfg_t_param_act_to_rdwr 0x0					
caltiming1 0x80	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved		cfg_t_param_rd_to_wr_diff_chip 0x0						cfg_t_param_rd_to_wr 0x0						cfg_t_param_rd_to_rd_diff_bg 0x0	
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cfg_t_param_rd_to_rd_diff_bg 0x0				cfg_t_param_rd_to_rd_diff_chip 0x0						cfg_t_param_rd_to_rd 0x0					
caltiming2 0x84	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved		cfg_t_param_wr_to_wr_diff_chip 0x0						cfg_t_param_wr_to_wr 0x0						cfg_t_param_rd_ap_to_valid 0x0	
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cfg_t_param_rd_ap_to_valid 0x0				cfg_t_param_rd_to_pch 0x0						cfg_t_param_rd_to_wr_diff_bg 0x0					
caltiming3 0x88	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved		cfg_t_param_wr_to_pch 0x0						cfg_t_param_wr_to_rd_diff_bg 0x0						cfg_t_param_wr_to_rd_diff_chip 0x0	
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cfg_t_param_wr_to_rd_diff_chip 0x0				cfg_t_param_wr_to_rd 0x0						cfg_t_param_wr_to_wr_diff_bg 0x0					
caltiming4 0x8C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cfg_t_param_pdn_to_valid 0x0						cfg_t_param_arf_to_valid 0x0								cfg_t_param_pch_all_to_valid 0x0	
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cfg_t_param_pch_all_to_valid 0x0				cfg_t_param_pch_to_valid 0x0						cfg_t_param_wr_ap_to_valid 0x0					

Register Address Offset	Bit Fields															
caltiming5 0x90	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved												cfg_t_param_srf_to_zq_cal 0x0			
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cfg_t_param_srf_to_zq_cal 0x0						cfg_t_param_srf_to_valid 0x0									
caltiming6 0x94	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved			cfg_t_param_pdn_period 0x0												
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cfg_t_param_pdn_period 0x0			cfg_t_param_arf_period 0x0												
caltiming7 0x98	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved		cfg_t_param_mps_to_valid 0x0										cfg_t_param_mrs_to_valid 0x0			
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cfg_t_param_zqcs_to_valid 0x0						cfg_t_param_zqcl_to_valid 0x0									
caltiming8 0x9C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved				cfg_t_param_mmr_cmd_to_valid 0x0								cfg_t_param_rld3_multibank_ref_delay 0x0			cfg_t_param_mps_exit_cke_to_cs 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cfg_t_param_mps_exit_cke_to_cs 0x0			cfg_t_param_mps_exit_cs_to_cke 0x0			cfg_t_param_mpr_to_valid 0x0						cfg_t_param_mrr_to_valid 0x0			
caltiming9 0xA0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								cfg_t_param_4_act_to_act 0x0							

Register Address Offset	Bit Fields															
caltiming10 0xA4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								cfg_t_param_16_act_to_act 0x0							
dramaddrw 0xA8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved													cfg_cs_addr_width 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cfg_bank_group_addr_width 0x0		cfg_bank_addr_width 0x0				cfg_row_addr_width 0x0					cfg_col_addr_width 0x0				
sideband0 0xAC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															mr_cmd_trigger 0x0
sideband1 0xB0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved												mmr_refresh_req 0x0			
sideband2 0xB4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															mmr_zqcal_long_req 0x0
sideband3 0xB8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															mmr_zqcal_short_req 0x0

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
sideband4 0xBC	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved												mmr_self_rfsh_req 0x0			
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
sideband5 0xC0	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															mmr_dpd_mps_req 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
sideband6 0xC4	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															mr_cmd_ack 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
sideband7 0xC8	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															mmr_refresh_ack 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
sideband8 0xCC	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															mmr_zqcal_ack 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
sideband9 0xD0	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															mmr_self_rfsh_ack 0x0

Register Address Offset	Bit Fields															
sideband10 0xD4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															mmr_dpd_mps_ack 0x0
sideband11 0xD8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															mmr_auto_pd_ack 0x0
sideband12 0xDC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved									mr_cmd_rank 0x0				mr_cmd_type 0x0		
sideband13 0xE0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	mr_cmd_opcode 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	mr_cmd_opcode 0x0															
sideband14 0xE4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	mmr_refresh_bank 0x0															
sideband15 0xE8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved												mmr_stall_rank 0x0			

Register Address Offset	Bit Fields															
dramsts 0xEC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														phy_cal_fail 0x0	phy_cal_success 0x0
dbgdone 0xF0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															dbg_done 0x0
dbgsignals 0xF4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	dbg_signals_out 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	dbg_signals_out 0x0															
dbgreset 0xF8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														counter_one_reset 0x0	counter_zero_reset 0x0
dbgmatch 0xFC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	counter_one 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	counter_zero 0x0															
counter0mask 0x100	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	counter_zero_mask 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	counter_zero_mask 0x0															

Register Address Offset	Bit Fields															
counter1mask 0x104	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	counter_one_mask															
	0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	counter_one_mask															
	0x0															
counter0match 0x108	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	counter_zero_match															
	0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	counter_zero_match															
	0x0															
counter1match 0x10C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	counter_one_match															
	0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	counter_one_match															
	0x0															
niosreserve0 0x110	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	nios_reserve0															
	0x0															
niosreserve1 0x114	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	nios_reserve1															
	0x0															
niosreserve2 0x118	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	nios_reserve2															
	0x0															

dbgcfg0

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA000

Offset: 0x0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							cfg_dbg_mode 0x0				cfg_cmd_driver_sel 0x0	cfg_loopback_en 0x0	cfg_mmr_driver_sel 0x0	cfg_prbs_ctrl_sel 0x0	cfg_wdata_driver_sel 0x0

dbgcfg0 Fields

Bit	Name	Description	Access	Reset
8:5	cfg_dbg_mode	TBD	RW	0x0
4	cfg_cmd_driver_sel	TBD	RW	0x0
3	cfg_loopback_en	TBD	RW	0x0
2	cfg_mmr_driver_sel	TBD	RW	0x0
1	cfg_prbs_ctrl_sel	TBD	RW	0x0
0	cfg_wdata_driver_sel	TBD	RW	0x0

dbgcfg1

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA004

Offset: 0x4

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cfg_dbg_ctrl 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cfg_dbg_ctrl 0x0															

dbgcfg1 Fields

Bit	Name	Description	Access	Reset
31:0	cfg_dbg_ctrl	TBD	RW	0x0

dbgcfg2

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA008

Offset: 0x8

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cfg_bist_cmd0_u 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cfg_bist_cmd0_u 0x0															

dbgcfg2 Fields

Bit	Name	Description	Access	Reset
31:0	cfg_bist_cmd0_u	TBD	RW	0x0

dbgcfg3

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA00C

Offset: 0xC

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cfg_bist_cmd0_l 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cfg_bist_cmd0_l 0x0															

dbgcfg3 Fields

Bit	Name	Description	Access	Reset
31:0	cfg_bist_cmd0_l	TBD	RW	0x0

dbgcfg4

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA010

Offset: 0x10

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cfg_bist_cmd1_u 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cfg_bist_cmd1_u 0x0															

dbgcfg4 Fields

Bit	Name	Description	Access	Reset
31:0	cfg_bist_cmd1_u	TBD	RW	0x0

dbgcfg5

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA014

Offset: 0x14

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cfg_bist_cmd1_l 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cfg_bist_cmd1_l 0x0															

dbgcfg5 Fields

Bit	Name	Description	Access	Reset
31:0	cfg_bist_cmd1_l	TBD	RW	0x0

dbgcfg6

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA018

Offset: 0x18

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cfg_dbg_out_sel 0x0															

dbgcfg6 Fields

Bit	Name	Description	Access	Reset
15:0	cfg_dbg_out_sel	Select which debug signals sent out for observation	RW	0x0

reserve0

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA01C

Offset: 0x1C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cfg_reserve0 0x0															

reserve0 Fields

Bit	Name	Description	Access	Reset
15:0	cfg_reserve0	General purpose reserve register	RW	0x0

reserve1

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA020

Offset: 0x20

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cfg_reserve1 0x0															

reserve1 Fields

Bit	Name	Description	Access	Reset
15:0	cfg_reserve1	General purpose reserve register	RW	0x0

reserve2

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA024

Offset: 0x24

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cfg_reserve2 0x0															

reserve2 Fields

Bit	Name	Description	Access	Reset
15:0	cfg_reserve2	General purpose reserve register	RW	0x0

ctrlcfg0

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA028

Offset: 0x28

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved			cfg_dbc2_burst_length 0x0					cfg_dbc1_burst_length 0x0					cfg_dbc0_burst_length 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cfg_dbc0_burst_length 0x0			cfg_ctrl_burst_length 0x0				cfg_ac_pos 0x0	cfg_dimm_type 0x0				cfg_mem_type 0x0			

ctrlcfg0 Fields

Bit	Name	Description	Access	Reset
28:24	cfg_dbc2_burst_length	Configures burst length for DBC2. Legal values are valid for JEDEC allowed DRAM values for the DRAM selected in cfg_type. For DDR3 and DDR4, this should be programmed with 8 (binary "01000"), for RLDRAM III it can be programmed with 2 or 4 or 8	RW	0x0
23:19	cfg_dbc1_burst_length	Configures burst length for DBC1. Legal values are valid for JEDEC allowed DRAM values for the DRAM selected in cfg_type. For DDR3 and DDR4, this should be programmed with 8 (binary "01000"), for RLDRAM III it can be programmed with 2 or 4 or 8	RW	0x0
18:14	cfg_dbc0_burst_length	Configures burst length for DBC0. Legal values are valid for JEDEC allowed DRAM values for the DRAM selected in cfg_type. For DDR3 and DDR4, this should be programmed with 8 (binary "01000"), for RLDRAM III it can be programmed with 2 or 4 or 8	RW	0x0
13:9	cfg_ctrl_burst_length	Configures burst length for control path. Legal values are valid for JEDEC allowed DRAM values for the DRAM selected in cfg_type. For DDR3 and DDR4, this should be programmed with 8 (binary "01000"), for RLDRAM III it can be programmed with 2 or 4 or 8	RW	0x0

Bit	Name	Description	Access	Reset
8:7	cfg_ac_pos	Specify C/A (command/address) pin position. 2	RW	0x0
6:4	cfg_dimm_type	Selects dimm type. Program this field with one of the following binary values, "3	RW	0x0
3:0	cfg_mem_type	Selects memory type. Program this field with one of the following binary values, "0000" for DDR3 SDRAM, "0001" for DDR4 SDRAM, and "0011" for RLDRAM3.	RW	0x0

ctrlcfg1

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA02C

Offset: 0x2C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	cfg_dbc3_enable_dm 0x0	cfg_dbc2_enable_dm 0x0	cfg_dbc1_enable_dm 0x0	cfg_dbc0_enable_dm 0x0	cfg_ctrl_enable_dm 0x0	cfg_dqstrk_en 0x0	cfg_starve_limit 0x0							cfg_reorder_read 0x0	cfg_dbc3_reorder_rdata 0x0	cfg_dbc2_reorder_rdata 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cfg_dbc1_reorder_rdata 0x0	cfg_dbc0_reorder_rdata 0x0	cfg_ctrl_reorder_rdata 0x0	cfg_reorder_data 0x0	cfg_dbc3_enable_ecc 0x0	cfg_dbc2_enable_ecc 0x0	cfg_dbc1_enable_ecc 0x0	cfg_dbc0_enable_ecc 0x0	cfg_ctrl_enable_ecc 0x0	cfg_addr_order 0x0		cfg_dbc3_burst_length 0x0				

ctrlcfg1 Fields

Bit	Name	Description	Access	Reset
30	cfg_dbc3_enable_dm	Set to a one to enable DRAM operation if DM pins are connected.	RW	0x0
29	cfg_dbc2_enable_dm	Set to a one to enable DRAM operation if DM pins are connected.	RW	0x0
28	cfg_dbc1_enable_dm	Set to a one to enable DRAM operation if DM pins are connected.	RW	0x0
27	cfg_dbc0_enable_dm	Set to a one to enable DRAM operation if DM pins are connected.	RW	0x0
26	cfg_ctrl_enable_dm	Set to a one to enable DRAM operation if DM pins are connected.	RW	0x0
25	cfg_dqstrk_en	Enables DQS tracking in the PHY.	RW	0x0
24:19	cfg_starve_limit	Specifies the number of DRAM burst transactions an individual transaction will allow to reorder ahead of it before its priority is raised in the memory controller.	RW	0x0
18	cfg_reorder_read	This bit controls whether the controller can re-order read command to. 1	RW	0x0
17	cfg_dbc3_reorder_rdata	This bit controls whether the controller need to re-order the read return data.	RW	0x0
16	cfg_dbc2_reorder_rdata	This bit controls whether the controller need to re-order the read return data.	RW	0x0
15	cfg_dbc1_reorder_rdata	This bit controls whether the controller need to re-order the read return data.	RW	0x0
14	cfg_dbc0_reorder_rdata	This bit controls whether the controller need to re-order the read return data.	RW	0x0
13	cfg_ctrl_reorder_rdata	This bit controls whether the controller need to re-order the read return data.	RW	0x0

Bit	Name	Description	Access	Reset
12	cfg_reorder_data	This bit controls whether the controller can re-order operations to optimize SDRAM bandwidth. It should generally be set to a one.	RW	0x0
11	cfg_dbc3_enable_ecc	Enable the generation and checking of ECC.	RW	0x0
10	cfg_dbc2_enable_ecc	Enable the generation and checking of ECC.	RW	0x0
9	cfg_dbc1_enable_ecc	Enable the generation and checking of ECC.	RW	0x0
8	cfg_dbc0_enable_ecc	Enable the generation and checking of ECC.	RW	0x0
7	cfg_ctrl_enable_ecc	Enable the generation and checking of ECC.	RW	0x0
6:5	cfg_addr_order	Selects the order for address interleaving. Programming this field with different values gives different mappings between the AXI or Avalon-MM address and the SDRAM address. Program this field with the following binary values to select the ordering. "00" - chip, row, bank(BG, BA), column; "01" - chip, bank(BG, BA), row, column; "10"-row, chip, bank(BG, BA), column;	RW	0x0
4:0	cfg_dbc3_burst_length	Configures burst length for DBC3. Legal values are valid for JEDEC allowed DRAM values for the DRAM selected in cfg_type. For DDR3 abd DDR4, this should be programmed with 8 (binary "01000"), for RLDRAM III it can be programmed with 2 or 4 or 8	RW	0x0

ctrlcfg2

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA030

Offset: 0x30

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved					cfg_dbc3_pipe_lat 0x0			cfg_dbc2_pipe_lat 0x0			cfg_dbc1_pipe_lat 0x0			cfg_dbc0_pipe_lat 0x0	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cfg_dbc0_pipe_lat 0x0	cfg_dbc2ctrl_sel 0x0		cfg_dbc3_ctrl_sel 0x0	cfg_dbc2_ctrl_sel 0x0	cfg_dbc1_ctrl_sel 0x0	cfg_dbc0_ctrl_sel 0x0	cfg_ctrl2dbc_switch1 0x0		cfg_ctrl2dbc_switch0 0x0		cfg_dbc3_output_regd 0x0	cfg_dbc2_output_regd 0x0	cfg_dbc1_output_regd 0x0	cfg_dbc0_output_regd 0x0	cfg_ctrl_output_regd 0x0

ctrlcfg2 Fields

Bit	Name	Description	Access	Reset
26:24	cfg_dbc3_pipe_lat	Specifies in number of controller clock cycles the latency of pipelining the signals from control path to DBC3	RW	0x0
23:21	cfg_dbc2_pipe_lat	Specifies in number of controller clock cycles the latency of pipelining the signals from control path to DBC2	RW	0x0
20:18	cfg_dbc1_pipe_lat	Specifies in number of controller clock cycles the latency of pipelining the signals from control path to DBC1	RW	0x0
17:15	cfg_dbc0_pipe_lat	Specifies in number of controller clock cycles the latency of pipelining the signals from control path to DBC0	RW	0x0

Bit	Name	Description	Access	Reset
14:13	cfg_dbc2ctrl_sel	Specifies which DBC is driven by the local control path. 2	RW	0x0
12	cfg_dbc3_ctrl_sel	DBC3 - control path select. 1	RW	0x0
11	cfg_dbc2_ctrl_sel	DBC2 - control path select. 1	RW	0x0
10	cfg_dbc1_ctrl_sel	DBC1 - control path select. 1	RW	0x0
9	cfg_dbc0_ctrl_sel	DBC0 - control path select. 1	RW	0x0
8:7	cfg_ctrl2dbc_switch1	Select of the MUX ctrl2dbc_switch1. 2	RW	0x0
6:5	cfg_ctrl2dbc_switch0	Select of the MUX ctrl2dbc_switch0. 2	RW	0x0
4	cfg_dbc3_output_regd	Set to one to register the HMC command output. Set to 0 to disable it.	RW	0x0
3	cfg_dbc2_output_regd	Set to one to register the HMC command output. Set to 0 to disable it.	RW	0x0
2	cfg_dbc1_output_regd	Set to one to register the HMC command output. Set to 0 to disable it.	RW	0x0
1	cfg_dbc0_output_regd	Set to one to register the HMC command output. Set to 0 to disable it.	RW	0x0
0	cfg_ctrl_output_regd	Set to one to register the HMC command output. Set to 0 to disable it.	RW	0x0

ctrlcfg3

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA034

Offset: 0x34

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	cfg_rld3_multibank_mode 0x0			cfg_geardn_en 0x0	cfg_open_page_en 0x0	cfg_arbiter_type 0x0	cfg_dbc3_dualport_en 0x0	cfg_dbc2_dualport_en 0x0	cfg_dbc1_dualport_en 0x0	cfg_dbc0_dualport_en 0x0	cfg_ctrl_dualport_en 0x0	cfg_dbc3_in_protocol 0x0	cfg_dbc2_in_protocol 0x0	cfg_dbc1_in_protocol 0x0	cfg_dbc0_in_protocol 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cfg_ctrl_in_protocol 0x0	cfg_dbc3_cmd_rate 0x0			cfg_dbc2_cmd_rate 0x0			cfg_dbc1_cmd_rate 0x0			cfg_dbc0_cmd_rate 0x0			cfg_ctrl_cmd_rate 0x0		

ctrlcfg3 Fields

Bit	Name	Description	Access	Reset
30:28	cfg_rld3_multibank_mode	Multibank setting, specific for RLDRAM3. Set this to: - 3	RW	0x0
27	cfg_geardn_en	Set to 1 to enable the gear down mode for DDR4	RW	0x0
26	cfg_open_page_en	Set to 1 to enable the open page policy when command reordering is disabled (cfg_cmd_reorder = 0). This bit does not matter when cfg_cmd_reorder is 1.	RW	0x0
25	cfg_arbiter_type	Indicates controller arbiter operating mode. Set this to: - 1	RW	0x0
24	cfg_dbc3_dualport_en	Enable the second data port for RLDRAM3 only (BL=2 or 4)	RW	0x0
23	cfg_dbc2_dualport_en	Enable the second data port for RLDRAM3 only (BL=2 or 4)	RW	0x0
22	cfg_dbc1_dualport_en	Enable the second data port for RLDRAM3 only (BL=2 or 4)	RW	0x0
21	cfg_dbc0_dualport_en	Enable the second data port for RLDRAM3 only (BL=2 or 4)	RW	0x0

Bit	Name	Description	Access	Reset
20	cfg_ctrl_dualport_en	Enable the second command port for RLDRAM3 only (BL=2 or 4)	RW	0x0
19	cfg_dbc3_in_protocol	1	RW	0x0
18	cfg_dbc2_in_protocol	1	RW	0x0
17	cfg_dbc1_in_protocol	1	RW	0x0
16	cfg_dbc0_in_protocol	1	RW	0x0
15	cfg_ctrl_in_protocol	1	RW	0x0
14:12	cfg_dbc3_cmd_rate	3	RW	0x0
11:9	cfg_dbc2_cmd_rate	3	RW	0x0
8:6	cfg_dbc1_cmd_rate	3	RW	0x0
5:3	cfg_dbc0_cmd_rate	3	RW	0x0
2:0	cfg_ctrl_cmd_rate	3	RW	0x0

ctrlcfg4

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA038

Offset: 0x38

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cfg_dbc3_slot_offset 0x0		cfg_dbc2_slot_offset 0x0		cfg_dbc1_slot_offset 0x0		cfg_dbc0_slot_offset 0x0		cfg_ctrl_slot_offset 0x0		cfg_dbc3_slot_rotate_en 0x0			cfg_dbc2_slot_rotate_en 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cfg_dbc1_slot_rotate_en 0x0			cfg_dbc0_slot_rotate_en 0x0			cfg_ctrl_slot_rotate_en 0x0			cfg_pingpong_mode 0x0		cfg_tile_id 0x0				

ctrlcfg4 Fields

Bit	Name	Description	Access	Reset
31:30	cfg_dbc3_slot_offset	Enables afi information to be offset by numbers of FR cycles. Affected afi signal is afi_rdata_en, afi_rdata_en_full, afi_wdata_valid, afi_dqs_burst, afi_mrnk_write and afi_mrnk_read. Set this to: - 2	RW	0x0
29:28	cfg_dbc2_slot_offset	Enables afi information to be offset by numbers of FR cycles. Affected afi signal is afi_rdata_en, afi_rdata_en_full, afi_wdata_valid, afi_dqs_burst, afi_mrnk_write and afi_mrnk_read. Set this to: - 2	RW	0x0
27:26	cfg_dbc1_slot_offset	Enables afi information to be offset by numbers of FR cycles. Affected afi signal is afi_rdata_en, afi_rdata_en_full, afi_wdata_valid, afi_dqs_burst, afi_mrnk_write and afi_mrnk_read. Set this to: - 2	RW	0x0
25:24	cfg_dbc0_slot_offset	Enables afi information to be offset by numbers of FR cycles. Affected afi signal is afi_rdata_en, afi_rdata_en_full, afi_wdata_valid, afi_dqs_burst, afi_mrnk_write and afi_mrnk_read. Set this to: - 2	RW	0x0

Bit	Name	Description	Access	Reset
23:22	cfg_ctrl_slot_offset	Enables afi information to be offset by numbers of FR cycles. Affected afi signal is afi_rdata_en, afi_rdata_en_full, afi_wdata_valid, afi_dqs_burst, afi_mrnk_write and afi_mrnk_read. Set this to: - 2	RW	0x0
21:19	cfg_dbc3_slot_rotate_en	DBC3 slot rotate enable: bit[0] controls write, 1	RW	0x0
18:16	cfg_dbc2_slot_rotate_en	DBC2 slot rotate enable: bit[0] controls write, 1	RW	0x0
15:13	cfg_dbc1_slot_rotate_en	DBC1 slot rotate enable: bit[0] controls write, 1	RW	0x0
12:10	cfg_dbc0_slot_rotate_en	DBC0 slot rotate enable: bit[0] controls write, 1	RW	0x0
9:7	cfg_ctrl_slot_rotate_en	Cmd slot rotate enable: bit[0] controls write, 1	RW	0x0
6:5	cfg_pingpong_mode	Ping Pong mode: 2	RW	0x0
4:0	cfg_tile_id	Tile ID	RW	0x0

ctrlcfg5

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA03C

Offset: 0x3C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			cfg_dbc3_rc_en 0x0	cfg_dbc2_rc_en 0x0	cfg_dbc1_rc_en 0x0	cfg_dbc0_rc_en 0x0	cfg_ctrl_rc_en 0x0	cfg_row_cmd_slot 0x0				cfg_col_cmd_slot 0x0			

ctrlcfg5 Fields

Bit	Name	Description	Access	Reset
12	cfg_dbc3_rc_en	Set to 1 to enable the rate conversion. It converts QR input from core to HR inside HMC	RW	0x0
11	cfg_dbc2_rc_en	Set to 1 to enable the rate conversion. It converts QR input from core to HR inside HMC	RW	0x0
10	cfg_dbc1_rc_en	Set to 1 to enable the rate conversion. It converts QR input from core to HR inside HMC	RW	0x0
9	cfg_dbc0_rc_en	Set to 1 to enable the rate conversion. It converts QR input from core to HR inside HMC	RW	0x0
8	cfg_ctrl_rc_en	Set to 1 to enable the rate conversion. It converts QR input from core to HR inside HMC	RW	0x0
7:4	cfg_row_cmd_slot	Specify the row cmd slot. One hot encoding.	RW	0x0
3:0	cfg_col_cmd_slot	Specify the col cmd slot. One hot encoding.	RW	0x0

ctrlcfg6

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA040

Offset: 0x40

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cfg_cs_chip 0x0															

ctrlcfg6 Fields

Bit	Name	Description	Access	Reset
15:0	cfg_cs_chip	Chip select mapping scheme. Mapping seperated into 4 sections: [CS3] [CS2][CS1][CS0] Each section consists of 4 bits to indicate which CS_n signal should be active when command goes to current CS. Eg: if we set to 16	RW	0x0

ctrlcfg7

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA044

Offset: 0x44

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	cfg_wb_reserved_entry 0x0							cfg_rb_reserved_entry 0x0							cfg_clkgating_en 0x0

ctrlcfg7 Fields

Bit	Name	Description	Access	Reset
14:8	cfg_wb_reserved_entry	Specify how many enties are reserved in write buffer before almost full is asserted	RW	0x0
7:1	cfg_rb_reserved_entry	Specify how many enties are reserved in read buffer before almost full is asserted	RW	0x0
0	cfg_clkgating_en	Set to 1 to enable the clock gating. The clock is shut off for the whole HMC	RW	0x0

ctrlcfg8

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA048

Offset: 0x48

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													cfg_addr_mplx_en 0x0	cfg_ck_inv 0x0	cfg_3ds_en 0x0

ctrlcfg8 Fields

Bit	Name	Description	Access	Reset
2	cfg_addr_mplx_en	Setting to 1 enables RLD3 address mulplex mode	RW	0x0
1	cfg_ck_inv	Use to program CK polarity. 1	RW	0x0
0	cfg_3ds_en	Setting to 1 to enable #DS support for DDR4	RW	0x0

ctrlcfg9

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA04C

Offset: 0x4C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															cfg_dfx_bypass_en 0x0

ctrlcfg9 Fields

Bit	Name	Description	Access	Reset
0	cfg_dfx_bypass_en	Used for dft and timing characterization only. 1	RW	0x0

dramtiming0

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA050

Offset: 0x50

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													cfg_mem_clk_disable_entry_cycles 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cfg_mem_clk_disable_entry_cycles 0x0			cfg_power_saving_exit_cycles 0x0						cfg_tcl 0x0						

dramtiming0 Fields

Bit	Name	Description	Access	Reset
18:13	cfg_mem_clk_disable_entry_cycles	Set to a the number of clocks after the execution of an self-refresh to stop the clock. This register is generally set based on PHY design latency and should generally not be changed.	RW	0x0
12:7	cfg_power_saving_exit_cycles	The minimum number of cycles to stay in a low power state. This applies to both power down and self-refresh and should be set to the greater of tPD and tCKESR.	RW	0x0
6:0	cfg_tcl	Memory read latency.	RW	0x0

dramodt0

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA054

Offset: 0x54

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cfg_read_odt_chip 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cfg_write_odt_chip 0x0															

dramodt0 Fields

Bit	Name	Description	Access	Reset
31:16	cfg_read_odt_chip	ODT scheme setting for read command. Setting seperated into 4 sections: [CS3][CS2][CS1][CS0] Each section consists of 4 bits to indicate which chip should ODT be asserted when write occurs on current CS. Eg: if we set to 16	RW	0x0
15:0	cfg_write_odt_chip	ODT scheme setting for write command. Setting seperated into 4 sections: [CS3][CS2][CS1][CS0] Each section consists of 4 bits to indicate which chip should ODT be asserted when write occurs on current CS. Eg: if we set to 16	RW	0x0

dramodt1

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA058

Offset: 0x58

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								cfg_rd_odt_period 0x0						cfg_wr_odt_period 0x0	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cfg_wr_odt_period 0x0				cfg_rd_odt_on 0x0				cfg_wr_odt_on 0x0							

dramodt1 Fields

Bit	Name	Description	Access	Reset
23:18	cfg_rd_odt_period	Indicates number of memory clock cycle read ODT signal should stay asserted after rising edge	RW	0x0
17:12	cfg_wr_odt_period	Indicates number of memory clock cycle write ODT signal should stay asserted after rising edge	RW	0x0
11:6	cfg_rd_odt_on	Indicates number of memory clock cycle gap between read command and ODT signal rising edge	RW	0x0
5:0	cfg_wr_odt_on	Indicates number of memory clock cycle gap between write command and ODT signal rising edge	RW	0x0

sbcfg0

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA05C

Offset: 0x5C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cfg_rld3_refresh_seq1 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cfg_rld3_refresh_seq0 0x0															

sbcfg0 Fields

Bit	Name	Description	Access	Reset
31:16	cfg_rld3_refresh_seq1	Banks to Refresh for RLD3 in sequence 1. Must not be more than 4 banks	RW	0x0
15:0	cfg_rld3_refresh_seq0	Banks to Refresh for RLD3 in sequence 0. Must not be more than 4 banks	RW	0x0

sbcfg1

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA060

Offset: 0x60

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cfg_rld3_refresh_seq3 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cfg_rld3_refresh_seq2 0x0															

sbcfg1 Fields

Bit	Name	Description	Access	Reset
31:16	cfg_rld3_refresh_seq3	Banks to Refresh for RLD3 in sequence 3. Must not be more than 4 banks	RW	0x0
15:0	cfg_rld3_refresh_seq2	Banks to Refresh for RLD3 in sequence 2. Must not be more than 4 banks	RW	0x0

sbcfg2

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA064

Offset: 0x64

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								cfg_srf_entry_exit_block 0x0		cfg_srf_autoexit_en 0x0	cfg_user_rfsh_en 0x0	cfg_sb_cg_disable 0x0	cfg_mps_dqstrk_disable 0x0	cfg_mps_zqcal_disable 0x0	cfg_srf_zqcal_disable 0x0

sbcfg2 Fields

Bit	Name	Description	Access	Reset
7:6	cfg_srf_entry_exit_block	Blocking arbiter from issuing cmds for the 4 cases, 2	RW	0x0
5	cfg_srf_autoexit_en	Setting to 1 to enable controller to exit Self Refresh when new command is detected	RW	0x0
4	cfg_user_rfsh_en	Setting to 1 to enable user refresh	RW	0x0
3	cfg_sb_cg_disable	Set to 1 to disable mem_ck gating during self refresh and deep power down	RW	0x0
2	cfg_mps_dqstrk_disable	Set to 1 to disable DQS Tracking after Maximum Power Saving exit	RW	0x0

Bit	Name	Description	Access	Reset
1	cfg_mps_zqcal_disable	Set to 1 to disable ZQ Calibration after Maximum Power Saving exit	RW	0x0
0	cfg_srf_zqcal_disable	Set to 1 to disable ZQ Calibration after self refresh	RW	0x0

sbcfg3

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA068

Offset: 0x68

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved												cfg_sb_ddr4_mr3 0x0			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cfg_sb_ddr4_mr3 0x0															

sbcfg3 Fields

Bit	Name	Description	Access	Reset
19:0	cfg_sb_ddr4_mr3	This register stores the DDR4 MR3 Content	RW	0x0

sbcfg4

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA06C

Offset: 0x6C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved												cfg_sb_ddr4_mr4 0x0			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cfg_sb_ddr4_mr4 0x0															

sbcfg4 Fields

Bit	Name	Description	Access	Reset
19:0	cfg_sb_ddr4_mr4	This register stores the DDR4 MR4 Content	RW	0x0

sbcfg5

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA070

Offset: 0x70

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														cfg_period_dqstrk_ctrl_en 0x0	cfg_short_dqstrk_ctrl_en 0x0

sbcfg5 Fields

Bit	Name	Description	Access	Reset
1	cfg_period_dqstrk_ctrl_en	Set to 1 to enable controller to issue periodic DQS tracking	RW	0x0
0	cfg_short_dqstrk_ctrl_en	Set to 1 to enable controller controlled DQS short tracking, Set to 0 to enable sequencer controlled DQS short tracking	RW	0x0

sbcfg6

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA074

Offset: 0x74

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cfg_t_param_dqstrk_to_valid 0x0								cfg_t_param_dqstrk_to_valid_last 0x0							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cfg_period_dqstrk_interval 0x0															

sbcfg6 Fields

Bit	Name	Description	Access	Reset
31:24	cfg_t_param_dqstrk_to_valid	DQS Tracking Rd to Valid timing for Ranks other than the Last	RW	0x0
23:16	cfg_t_param_dqstrk_to_valid_last	DQS Tracking Rd to Valid timing for the last Rank	RW	0x0
15:0	cfg_period_dqstrk_interval	Inverval between two controller controlled periodic DQS tracking	RW	0x0

sbcfg7

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA078

Offset: 0x78

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									cfg_rfsh_warn_threshold 0x0						

sbcfg7 Fields

Bit	Name	Description	Access	Reset
6:0	cfg_rfsh_warn_threshold	Threshold to warn a refresh is needed within the number of controller clock cycles specified by the threshold	RW	0x0

caltiming0

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA07C

Offset: 0x7C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		cfg_t_param_act_to_act_diff_bg 0x0						cfg_t_param_act_to_act_diff_bank 0x0						cfg_t_param_act_to_act 0x0	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cfg_t_param_act_to_act 0x0				cfg_t_param_act_to_pch 0x0						cfg_t_param_act_to_rdwr 0x0					

caltiming0 Fields

Bit	Name	Description	Access	Reset
29:24	cfg_t_param_act_to_act_diff_bg	Active to activate timing on different bank groups, DDR4 only	RW	0x0
23:18	cfg_t_param_act_to_act_diff_bank	Active to activate timing on different banks, for DDR4 same bank group	RW	0x0
17:12	cfg_t_param_act_to_act	Active to activate timing on same bank	RW	0x0
11:6	cfg_t_param_act_to_pch	Active to precharge	RW	0x0
5:0	cfg_t_param_act_to_rdwr	Activate to Read/write command timing	RW	0x0

caltiming1

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA080

Offset: 0x80

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		cfg_t_param_rd_to_wr_diff_chip 0x0						cfg_t_param_rd_to_wr 0x0						cfg_t_param_rd_to_rd_diff_bg 0x0	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cfg_t_param_rd_to_rd_diff_bg 0x0				cfg_t_param_rd_to_rd_diff_chip 0x0						cfg_t_param_rd_to_rd 0x0					

caltiming1 Fields

Bit	Name	Description	Access	Reset
29:24	cfg_t_param_rd_to_wr_diff_chip	Read to write command timing on different chips	RW	0x0
23:18	cfg_t_param_rd_to_wr	Write to read command timing on same bank	RW	0x0
17:12	cfg_t_param_rd_to_rd_diff_bg	Read to read command timing on different chips	RW	0x0
11:6	cfg_t_param_rd_to_rd_diff_chip	Read to read command timing on different chips	RW	0x0
5:0	cfg_t_param_rd_to_rd	Read to read command timing on same bank	RW	0x0

caltiming2

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA084

Offset: 0x84

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		cfg_t_param_wr_to_wr_diff_chip 0x0						cfg_t_param_wr_to_wr 0x0						cfg_t_param_rd_ap_to_valid 0x0	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cfg_t_param_rd_ap_to_valid 0x0				cfg_t_param_rd_to_pch 0x0						cfg_t_param_rd_to_wr_diff_bg 0x0					

caltiming2 Fields

Bit	Name	Description	Access	Reset
29:24	cfg_t_param_wr_to_wr_diff_chip	Write to write command timing on different chips.	RW	0x0
23:18	cfg_t_param_wr_to_wr	Write to write command timing on same bank.	RW	0x0
17:12	cfg_t_param_rd_ap_to_valid	Read command with autoprecharge to data valid timing	RW	0x0
11:6	cfg_t_param_rd_to_pch	Read to precharge command timing	RW	0x0
5:0	cfg_t_param_rd_to_wr_diff_bg	Read to write command timing on different bank groups	RW	0x0

caltiming3

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA088

Offset: 0x88

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		cfg_t_param_wr_to_pch 0x0						cfg_t_param_wr_to_rd_diff_bg 0x0						cfg_t_param_wr_to_rd_diff_chip 0x0	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cfg_t_param_wr_to_rd_diff_chip 0x0				cfg_t_param_wr_to_rd 0x0				cfg_t_param_wr_to_wr_diff_bg 0x0							

caltiming3 Fields

Bit	Name	Description	Access	Reset
29:24	cfg_t_param_wr_to_pch	Write to precharge command timing.	RW	0x0
23:18	cfg_t_param_wr_to_rd_diff_bg	Write to read command timing on different bank groups	RW	0x0
17:12	cfg_t_param_wr_to_rd_diff_chip	Write to read command timing on different chips.	RW	0x0
11:6	cfg_t_param_wr_to_rd	Write to read command timing.	RW	0x0
5:0	cfg_t_param_wr_to_wr_diff_bg	Write to write command timing on different bank groups.	RW	0x0

caltiming4

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA08C

Offset: 0x8C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cfg_t_param_pdn_to_valid 0x0						cfg_t_param_arf_to_valid 0x0								cfg_t_param_pch_all_to_valid 0x0	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cfg_t_param_pch_all_to_valid 0x0				cfg_t_param_pch_to_valid 0x0						cfg_t_param_wr_ap_to_valid 0x0					

caltiming4 Fields

Bit	Name	Description	Access	Reset
31:26	cfg_t_param_pdn_to_valid	Power down to valid bank command window.	RW	0x0
25:18	cfg_t_param_arf_to_valid	Auto Refresh to valid DRAM command window.	RW	0x0
17:12	cfg_t_param_pch_all_to_valid	Precharge all to banks being ready for bank activation command.	RW	0x0
11:6	cfg_t_param_pch_to_valid	Precharge to valid command timing.	RW	0x0
5:0	cfg_t_param_wr_ap_to_valid	Write with autoprecharge to valid command timing.	RW	0x0

caltiming5

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA090

Offset: 0x90

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved												cfg_t_param_srf_to_zq_cal 0x0			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cfg_t_param_srf_to_zq_cal 0x0						cfg_t_param_srf_to_valid 0x0									

caltiming5 Fields

Bit	Name	Description	Access	Reset
19:10	cfg_t_param_srf_to_zq_cal	Self refresh to ZQ calibration window	RW	0x0
9:0	cfg_t_param_srf_to_valid	Self-refresh to valid bank command window.	RW	0x0

caltiming6

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA094

Offset: 0x94

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved			cfg_t_param_pdn_period 0x0												
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cfg_t_param_pdn_period 0x0			cfg_t_param_arf_period 0x0												

caltiming6 Fields

Bit	Name	Description	Access	Reset
28:13	cfg_t_param_pdn_period	Clock power down recovery period.	RW	0x0
12:0	cfg_t_param_arf_period	Auto-refresh period	RW	0x0

caltiming7

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA098

Offset: 0x98

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		cfg_t_param_mps_to_valid 0x0										cfg_t_param_mrs_to_valid 0x0			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cfg_t_param_zqcs_to_valid 0x0							cfg_t_param_zqcl_to_valid 0x0								

caltiming7 Fields

Bit	Name	Description	Access	Reset
29:20	cfg_t_param_mps_to_valid	Timing parameter for Maximum Power Saving to any valid command. tXMP	RW	0x0
19:16	cfg_t_param_mrs_to_valid	Mode Register Setting to valid	RW	0x0
15:9	cfg_t_param_zqcs_to_valid	Short ZQ calibration to valid	RW	0x0
8:0	cfg_t_param_zqcl_to_valid	Long ZQ calibration to valid	RW	0x0

caltiming8

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA09C

Offset: 0x9C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				cfg_t_param_mmr_cmd_to_valid 0x0								cfg_t_param_rld3_multibank_ref_delay 0x0			cfg_t_param_mps_exit_cke_to_cs 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cfg_t_param_mps_exit_cke_to_cs 0x0			cfg_t_param_mps_exit_cs_to_cke 0x0				cfg_t_param_mpr_to_valid 0x0					cfg_t_param_mrr_to_valid 0x0			

caltiming8 Fields

Bit	Name	Description	Access	Reset
27:20	cfg_t_param_mmr_cmd_to_valid	MMR cmd to valid delay	RW	0x0
19:17	cfg_t_param_rld3_multibank_ref_delay	RLD3 Refresh to Refresh Delay for all sequences	RW	0x0
16:13	cfg_t_param_mps_exit_cke_to_cs	Timing parameter for exit Maximum Power Saving. Timing requirement for CKE de-assertion vs CS de-assertion. tMPX_LH	RW	0x0
12:9	cfg_t_param_mps_exit_cs_to_cke	Timing parameter for exit Maximum Power Saving. Timing requirement for CS assertion vs CKE de-assertion. tMPX_S	RW	0x0
8:4	cfg_t_param_mpr_to_valid	Timing parameter for Multi Purpose Register Read to any valid command	RW	0x0
3:0	cfg_t_param_mrr_to_valid	Timing parameter for Mode Register Read to any valid command	RW	0x0

caltiming9

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA0A0

Offset: 0xA0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								cfg_t_param_4_act_to_act 0x0							

caltiming9 Fields

Bit	Name	Description	Access	Reset
7:0	cfg_t_param_4_act_to_act	The four-activate window timing parameter.	RW	0x0

caltiming10

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA0A4

Offset: 0xA4

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								cfg_t_param_16_act_to_act 0x0							

caltiming10 Fields

Bit	Name	Description	Access	Reset
7:0	cfg_t_param_16_act_to_act	The 16-activate window timing parameter (RLD3).	RW	0x0

dramaddrw

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA0A8

Offset: 0xA8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													cfg_cs_addr_width 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cfg_bank_group_addr_width 0x0		cfg_bank_addr_width 0x0				cfg_row_addr_width 0x0					cfg_col_addr_width 0x0				

dramaddrw Fields

Bit	Name	Description	Access	Reset
18:16	cfg_cs_addr_width	The number of chip select address bits for the memory devices in your memory interface.	RW	0x0
15:14	cfg_bank_group_addr_width	The number of bank group address bits for the memory devices in your memory interface.	RW	0x0

Bit	Name	Description	Access	Reset
13:10	cfg_bank_addr_width	The number of bank address bits for the memory devices in your memory interface.	RW	0x0
9:5	cfg_row_addr_width	The number of row address bits for the memory devices in your memory interface.	RW	0x0
4:0	cfg_col_addr_width	The number of column address bits for the memory devices in your memory interface.	RW	0x0

sideband0

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA0AC

Offset: 0xAC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															mr_cmd_trigger 0x0

sideband0 Fields

Bit	Name	Description	Access	Reset
0	mr_cmd_trigger	Write to 1 to trigger the execution of the mode register command. It	RW	0x0

sideband1

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA0B0

Offset: 0xB0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												mmr_refresh_req 0x0			

sideband1 Fields

Bit	Name	Description	Access	Reset
3:0	mmr_refresh_req	When asserted, indicates Refresh request to the specific rank. Each bit corresponds to each rank. Controller clear this bit to	RW	0x0

sideband2

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA0B4

Offset: 0xB4

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															mmr_zqcal_long_req 0x0

sideband2 Fields

Bit	Name	Description	Access	Reset
0	mmr_zqcal_long_req	When asserted, indicates long ZQ cal request. This bit is write clear	RW	0x0

sideband3

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA0B8

Offset: 0xB8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															mmr_zqcal_short_req 0x0

sideband3 Fields

Bit	Name	Description	Access	Reset
0	mmr_zqcal_short_req	When asserted, indicates short ZQ cal request. This bit is write clear	RW	0x0

sideband4

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA0BC

Offset: 0xBC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												mmr_self_rfsh_req 0x0			

sideband4 Fields

Bit	Name	Description	Access	Reset
3:0	mmr_self_rfsh_req	When asserted, indicates self refresh request to the specific rank. Each bit corresponds to each rank. These bits are write clear	RW	0x0

sideband5

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA0C0

Offset: 0xC0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															mmr_dpd_mps_req 0x0

sideband5 Fields

Bit	Name	Description	Access	Reset
0	mmr_dpd_mps_req	When asserted, indicates deep power down or max power saving request. This bit is write clear	RW	0x0

sideband6

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA0C4

Offset: 0xC4

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															mr_cmd_ack 0x0

sideband6 Fields

Bit	Name	Description	Access	Reset
0	mr_cmd_ack	Acknowledge to mode register command	RO	0x0

sideband7

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA0C8

Offset: 0xC8

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															mmr_refresh_ack 0x0

sideband7 Fields

Bit	Name	Description	Access	Reset
0	mmr_refresh_ack	Acknowledge to indicate refresh is in progress	RO	0x0

sideband8

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA0CC

Offset: 0xCC

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															mmr_zqcal_ack 0x0

sideband8 Fields

Bit	Name	Description	Access	Reset
0	mmr_zqcal_ack	Acknowledge to indicate ZQCAL is progress	RO	0x0

sideband9

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA0D0

Offset: 0xD0

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															mmr_self_rfsh_ack 0x0

sideband9 Fields

Bit	Name	Description	Access	Reset
0	mmr_self_rfsh_ack	Acknowledge to indicate self refresh is progress	RO	0x0

sideband10

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA0D4

Offset: 0xD4

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															mmr_dpd_mps_ack 0x0

sideband10 Fields

Bit	Name	Description	Access	Reset
0	mmr_dpd_mps_ack	Acknowledge to indicate deep power down/max power saving is in progress	RO	0x0

sideband11

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA0D8

Offset: 0xD8

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															mmr_auto_pd_ack 0x0

sideband11 Fields

Bit	Name	Description	Access	Reset
0	mmr_auto_pd_ack	Acknowledge to indicate auto power down is in progress	RO	0x0

sideband12

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA0DC

Offset: 0xDC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									mr_cmd_rank 0x0				mr_cmd_type 0x0		

sideband12 Fields

Bit	Name	Description	Access	Reset
6:3	mr_cmd_rank	Indicates which rank the mode register command is intended to.	RW	0x0
2:0	mr_cmd_type	Indicates the type of Mode Register Command	RW	0x0

sideband13

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA0E0

Offset: 0xE0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
mr_cmd_opcode 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
mr_cmd_opcode 0x0															

sideband13 Fields

Bit	Name	Description	Access	Reset
31:0	mr_cmd_opcode	[31:27] reserved. Register Command Opcode Information to be used for Register Command Reserved [31:10] falling edge CA. [9:0] rising edge CA DDR4 [26:24] C2:C0 [23] ACT [22:21] BG1:BG0 [20] Reserved [19:18] BA1:BA0 [17] A17 [16] RAS [15] CAS [14] WE [13:0] A13:A0 DDR3 [26:21] Reserved [20:18] BA2:BA0 [17] A17 [16] RAS [15] CAS [14] WE [13] Reserved [12:0] A12:A0 RLDRAM3 [26] Reserved [25:22] BA3:BA0 [21] REF [20] WE [19:0] A19:A0	RW	0x0

sideband14

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA0E4

Offset: 0xE4

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
mmr_refresh_bank 0x0															

sideband14 Fields

Bit	Name	Description	Access	Reset
15:0	mmr_refresh_bank	user refresh bank information, binary representation of bank address. Enables refresh to that bank address when requested	RW	0x0

sideband15

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA0E8

Offset: 0xE8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												mmr_stall_rank 0x0			

sideband15 Fields

Bit	Name	Description	Access	Reset
3:0	mmr_stall_rank	Setting to 1 to stall the corresponding rank	RW	0x0

dramsts

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA0EC

Offset: 0xEC

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														phy_cal_fail 0x0	phy_cal_success 0x0

dramsts Fields

Bit	Name	Description	Access	Reset
1	phy_cal_fail	This bit will be set to 1 if the PHY was unable to calibrate.	RO	0x0
0	phy_cal_success	This bit will be set to 1 if the PHY was able to successfully calibrate.	RO	0x0

dbgdone

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA0F0

Offset: 0xF0

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															dbg_done 0x0

dbgdone Fields

Bit	Name	Description	Access	Reset
0	dbg_done	Indicates the debug test is completed	RO	0x0

dbgsignals

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA0F4

Offset: 0xF4

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dbg_signals_out 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
dbg_signals_out 0x0															

dbgsignals Fields

Bit	Name	Description	Access	Reset
31:0	dbg_signals_out	Debug signals output	RO	0x0

dbgreset

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA0F8

Offset: 0xF8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														counter_one_reset 0x0	counter_zero_reset 0x0

dbgreset Fields

Bit	Name	Description	Access	Reset
1	counter_one_reset	Used for performance monitoring. Writing to this register resets the second counter. Note that this bit auto-clears after one clock cycle.	RW	0x0
0	counter_zero_reset	Used for performance monitoring. Writing to this register resets the first counter. Note that this bit auto-clears after one clock cycle.	RW	0x0

dbgmatch

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA0FC

Offset: 0xFC

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
counter_one 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
counter_zero 0x0															

dbgmatch Fields

Bit	Name	Description	Access	Reset
31:16	counter_one	counter value	RO	0x0
15:0	counter_zero	counter value	RO	0x0

counter0mask

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA100

Offset: 0x100

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
counter_zero_mask 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
counter_zero_mask 0x0															

counter0mask Fields

Bit	Name	Description	Access	Reset
31:0	counter_zero_mask	Performance monitoring register. This register is used to mask off the internal signals selected by the debug select byte to either examine a bit (and expect it to be a one or a zero) or to ignore the bit.	RW	0x0

counter1mask

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA104

Offset: 0x104

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
counter_one_mask 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
counter_one_mask 0x0															

counter1mask Fields

Bit	Name	Description	Access	Reset
31:0	counter_one_mask	Performance monitoring register. This register is used to mask off the internal signals selected by the debug select byte to either examine a bit (and expect it to be a one or a zero) or to ignore the bit.	RW	0x0

counter0match

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA108

Offset: 0x108

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
counter_zero_match 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
counter_zero_match 0x0															

counter0match Fields

Bit	Name	Description	Access	Reset
31:0	counter_zero_match	Counts events which happens during the sample window which counter_zero_mask was satisfied.	RW	0x0

counter1match

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA10C

Offset: 0x10C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
counter_one_match 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
counter_one_match 0x0															

counter1match Fields

Bit	Name	Description	Access	Reset
31:0	counter_one_match	Counts events which happens during the sample window which counter_one_mask was satisfied.	RW	0x0

niosreserve0

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA110

Offset: 0x110

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
nios_reserve0 0x0															

niosreserve0 Fields

Bit	Name	Description	Access	Reset
15:0	nios_reserve0	Reserved register0 for Nios	RW	0x0

niosreserve1

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA114

Offset: 0x114

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
nios_reserve1 0x0															

niosreserve1 Fields

Bit	Name	Description	Access	Reset
15:0	nios_reserve1	Reserved register1 for Nios	RW	0x0

niosreserve2

Module Instance	Base Address	Register Address
i_io48_hmc_mmr_io48_mmr	0xFFCFA000	0xFFCFA118

Offset: 0x118

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
nios_reserve2 0x0															

niosreserve2 Fields

Bit	Name	Description	Access	Reset
15:0	nios_reserve2	Reserved register2 for Nios	RW	0x0

ecc_hmc_ocp_slv_block Address Map

Module Instance	Base Address	End Address
ecc_hmc_ocp_slv_block	0xFFCFB000	0xFFCFDFFF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
IP_REV_ID on page 7-138	0x0	32	RO	0x0	IP_REV_ID
DDRIOCTRL on page 7-139	0x08	32	RW	0x0	DDR IO Control Register
DDRCALSTAT on page 7-140	0x00C	32	RO	0x0	DDR Calibration Register
MPR_0BEAT1 on page 7-140	0x010	32	RO	0x0	MPR register [31:0] for first beat
MPR_1BEAT1 on page 7-141	0x014	32	RO	0x0	MPR register [63:32] for first beat
MPR_2BEAT1 on page 7-142	0x018	32	RO	0x0	MPR register [95:64] for first beat

Register	Offset	Width	Access	Reset Value	Description
MPR_3BEAT1 on page 7-142	0x01C	32	RO	0x0	MPR register [127:96] for first beat
MPR_4BEAT1 on page 7-143	0x020	32	RO	0x0	MPR register [159:128] for first beat
MPR_5BEAT1 on page 7-144	0x024	32	RO	0x0	MPR register [191:160] for first beat
MPR_6BEAT1 on page 7-144	0x028	32	RO	0x0	MPR register [223:192] for first beat
MPR_7BEAT1 on page 7-145	0x02C	32	RO	0x0	MPR register [255:224] for first beat
MPR_8BEAT1 on page 7-146	0x030	32	RO	0x0	MPR register [287:256] for first beat
MPR_0BEAT2 on page 7-146	0x034	32	RO	0x0	MPR register [31:0] for second beat
MPR_1BEAT2 on page 7-147	0x038	32	RO	0x0	MPR register [63:32] for second beat
MPR_2BEAT2 on page 7-148	0x03C	32	RO	0x0	MPR register [95:64] for second beat
MPR_3BEAT2 on page 7-148	0x040	32	RO	0x0	MPR register [127:96] for second beat
MPR_4BEAT2 on page 7-149	0x044	32	RO	0x0	MPR register [159:128] for second beat
MPR_5BEAT2 on page 7-150	0x048	32	RO	0x0	MPR register [191:160] for second beat
MPR_6BEAT2 on page 7-150	0x04C	32	RO	0x0	MPR register [223:192] for second beat
MPR_7BEAT2 on page 7-151	0x050	32	RO	0x0	MPR register [255:224] for second beat
MPR_8BEAT2 on page 7-152	0x054	32	RO	0x0	MPR register [287:256] for second beat
AUTO_PRECHARGE on page 7-152	0x60	32	RW	0x0	auto-precharge bit

Register	Offset	Width	Access	Reset Value	Description
ECCCTRL1 on page 7-153	0x100	32	RW	0x0	ECC control 1. This bit is used to set the initialize the memory and ecc to a known value
ECCCTRL2 on page 7-154	0x104	32	RW	0x0	ECC control 2. This bit is used to set the initialize the memory and ecc to a known value
ERRINTEN on page 7-156	0x110	32	RW	0x0	Error Interrupt enable
ERRINTENS on page 7-157	0x114	32	RW	0x0	Error Interrupt set
ERRINTENR on page 7-158	0x118	32	RW	0x0	Error Interrupt reset.
INTMODE on page 7-160	0x11C	32	RW	0x0	Interrupt mode
INTSTAT on page 7-161	0x120	32	RW	0x0	Interrupt status
DIAGINTTEST on page 7-163	0x124	32	RW	0x0	Enable diagnostic errors
MODSTAT on page 7-166	0x128	32	RW	0x0	Counter feature status flag
DERRADDRA on page 7-168	0x12C	32	RO	0x0	Double-bit error address
SERRADDRA on page 7-168	0x130	32	RO	0x0	Single-bit error address
AUTOWB_CORRADDR on page 7-169	0x138	32	RO	0x0	This register shows the address of the current autoWB correction SBE.
SERRCNTREG on page 7-170	0x13C	32	RW	0x0	Maximum counter value for single-bit error interrupt
AUTOWB_DROP_CNTREG on page 7-171	0x140	32	RW	0x1	Maximum counter value for AUTOWB correction interrupt

Register	Offset	Width	Access	Reset Value	Description
ECC_REG2WRECCDATABUS on page 7-172	0x144	32	RW	0x0	ECC from register associated to data which will be written to the RAM
ECC_RDECCDATA2REGBUS	0x148	32	RO	0x0	ECC of data from RAM will be written to register
ECC_REG2RDECCDATABUS on page 7-173	0x14C	32	RW	0x0	ECC from register associated to RD data which will be written to hmc ecc
ECC_DIAGON on page 7-174	0x150	32	RW	0x0	Enable diagnostics access
ECC_DECSTAT on page 7-176	0x154	32	RW	0x0	Diagnostic decoder status
ECC_ERRGENADDR_0 on page 7-180	0x160	32	RO	0x0	Error address register
ECC_ERRGENADDR_1 on page 7-181	0x164	32	RO	0x0	Error address register
ECC_ERRGENADDR_2 on page 7-182	0x168	32	RO	0x0	Error address register
ECC_ERRGENADDR_3 on page 7-183	0x16C	32	RO	0x0	Error address register
ECC_REG2RDDATABUS_BEAT0 on page 7-184	0x170	32	RW	0x0	ECC Reg2Rddatabus_beat0
ECC_REG2RDDATABUS_BEAT1 on page 7-185	0x174	32	RW	0x0	ECC Reg2Rddatabus_beat1
ECC_REG2RDDATABUS_BEAT2 on page 7-187	0x178	32	RW	0x0	ECC Reg2Rddatabus_beat2
ECC_REG2RDDATABUS_BEAT3 on page 7-188	0x17C	32	RW	0x0	ECC Reg2Rddatabus_beat3

ecc_hmc_ocp_slv_block Summary

Base Address: 0xFFCFB000

Register Address Offset	Bit Fields															
ecc_hmc_ocp_slv_block																
IP_REV_ID 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SIREV 0x0															
DDRIOCTRL 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														IO_SIZE 0x0	
DDRCALSTAT 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															CAL 0x0
MPR_0BEAT1 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	MPR0 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	MPR0 0x0															
MPR_1BEAT1 0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	MPR32 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	MPR32 0x0															
MPR_2BEAT1 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	MPR64 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	MPR64 0x0															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	MPR96 0x0															
MPR_3BEAT1 0x1C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	MPR96 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	MPR128 0x0															
MPR_4BEAT1 0x20	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	MPR128 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	MPR160 0x0															
MPR_5BEAT1 0x24	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	MPR160 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	MPR192 0x0															
MPR_6BEAT1 0x28	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	MPR192 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	MPR224 0x0															
MPR_7BEAT1 0x2C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	MPR224 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	MPR256 0x0															
MPR_8BEAT1 0x30	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	MPR256 0x0															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MPR_0BEAT2	MPR0															
0x34	0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	MPR0															
	0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MPR_1BEAT2	MPR32															
0x38	0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	MPR32															
	0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MPR_2BEAT2	MPR64															
0x3C	0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	MPR64															
	0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MPR_3BEAT2	MPR96															
0x40	0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	MPR96															
	0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MPR_4BEAT2	MPR128															
0x44	0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	MPR128															
	0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MPR_5BEAT2	MPR160															
0x48	0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	MPR160															
	0x0															

Register Address Offset	Bit Fields															
MPR_6BEAT2 0x4C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	MPR192 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	MPR192 0x0															
MPR_7BEAT2 0x50	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	MPR224 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	MPR224 0x0															
MPR_8BEAT2 0x54	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	MPR256 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	MPR256 0x0															
AUTO_PRECHARGE 0x60	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															CTRL 0x0
ECCCTRL1 0x100	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															AUTOWB_CNT_RST 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							CNT_RST 0x0	Reserved							ECC_EN 0x0
ECCCTRL2 0x104	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															OVRW_RB_ECC_EN 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							RMW_EN 0x0	Reserved							AUTOWB_EN 0x0

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
ERRINTEN	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x110	Reserved													HMI_INTREN 0x0	DERRINTEN 0x0	SERRINTEN 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
ERRINTENS	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x114	Reserved													HMI_INTRS 0x0	DERRINTS 0x0	SERRINTS 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
ERRINTENR	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x118	Reserved													HMI_INTRR 0x0	DERRINTR 0x0	SERRINTR 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved							AFICAL_EN 0x0	Reserved							INTONCMP 0x0
INTMODE	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x11C	Reserved							EXT_ADDRPARITY_EN 0x0	Reserved							INTMODE 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved													DERRBUSFLG 0x0	ADDRPARFLG 0x0	ADDRMTCFLG 0x0
INTSTAT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x120	Reserved													HMI_PENA 0x0	DERRPENA 0x0	SERRPENA 0x0

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DIAGINTTEST 0x124	Reserved							TADDRPAR 0x0	Reserved							TADDRMTC 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							TDERRA 0x0	Reserved							TSERRA 0x0
MODSTAT 0x128	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							AUTOWB_DROP_FLG 0x0	Reserved							CMPFLGA 0x0
DERRADDRA 0x12C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	DADDRESS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	DADDRESS 0x0															
SERRADDRA 0x130	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	SADDRESS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SADDRESS 0x0															
AUTOWB_CORRADDR 0x138	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	SWBADDRESS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SWBADDRESS 0x0															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SERRCNTREG	SERRCNT															
0x13C	0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SERRCNT															
	0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AUTOWB_DROP_CNTREG	CNT															
0x140	0x1															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CNT															
	0x1															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_REG2WRECCDATABUS	ECC3BUS								ECC2BUS							
0x144	0x0								0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC1BUS								ECC0BUS							
	0x0								0x0							
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_REG2RDECCDATABUS	ECC3BUS								ECC2BUS							
0x14C	0x0								0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC1BUS								ECC0BUS							
	0x0								0x0							
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_DIAGON	Reserved															ECCDIAGON
0x150																0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														RDDIAGON	WRDIAGON
															0x0	0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_DECSTAT	Reserved															
0x154	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				DEC3DERRFLG	DEC2DERRFLG	DEC1DERRFLG	DEC0DERRFLG	DEC3ADDRFLG	DEC2ADDRFLG	DEC1ADDRFLG	DEC0ADDRFLG	DEC3SERRFLG	DEC2SERRFLG	DEC1SERRFLG	DEC0SERRFLG
					0x0	0x0	0x0	0x0	0x0	0x0	0x0	0x0	0x0	0x0	0x0	0x0

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_ERRGENADDR_0 0x160	ADDR 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ADDR 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_ERRGENADDR_1 0x164	ADDR 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ADDR 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_ERRGENADDR_2 0x168	ADDR 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ADDR 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_ERRGENADDR_3 0x16C	ADDR 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ADDR 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_REG2RDDATABUS_BEAT0 0x170	ECC3BUS 0x0								ECC2BUS 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC1BUS 0x0								ECC0BUS 0x0							
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_REG2RDDATABUS_BEAT1 0x174	ECC3BUS 0x0								ECC2BUS 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC1BUS 0x0								ECC0BUS 0x0							

Register Address Offset	Bit Fields															
ECC_REG2RDDATABUS_BEAT2 0x178	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC3BUS 0x0								ECC2BUS 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC1BUS 0x0								ECC0BUS 0x0							
ECC_REG2RDDATABUS_BEAT3 0x17C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC3BUS 0x0								ECC2BUS 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC1BUS 0x0								ECC0BUS 0x0							

IP_REV_ID

IDO Register

Module Instance	Base Address	Register Address
ecc_hmc_ocp_slv_block	0xFFCFB000	0xFFCFB000

Offset: 0x0

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SIREV 0x0															

IP_REV_ID Fields

Bit	Name	Description	Access	Reset
15:0	SIREV	IP Rev #These bits indicate the silicon revision number	RO	0x0

DDRIOCTRL

DDR IO Control Register

Module Instance	Base Address	Register Address
ecc_hmc_ocp_slv_block	0xFFCFB000	0xFFCFB008

Offset: 0x8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														IO_SIZE 0x0	

DDRIOCTRL Fields

Bit	Name	Description	Access	Reset
1:0	IO_SIZE	External Configuration of DDR IO size. This field indicates the width of the external DDR to which the hard memory cotnroller is interfacing. These bits are configured at the start to indicate the external DDRIO size. 0x0 = DDR IO x16. default value after reset 0x1 = DDR IO x32 0x2 = DDR IO x64	RW	0x0

DDRCALSTAT

DDR Calibration Register

Module Instance	Base Address	Register Address
ecc_hmc_ocp_slv_block	0xFFCFB000	0xFFCFB00C

Offset: 0xC

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															CAL 0x0

DDRCALSTAT Fields

Bit	Name	Description	Access	Reset
0	CAL	DDR calibration status. This field indicates the calibration status of the hard memory controller's I/O column. During this time the DDR memory is not available. 0x0: When clear, calibration is either on going, hasn't started or failed. 0x1: When set to 1, calibration has succeeded.	RO	0x0

MPR_0BEAT1

Multipurpose register [31:0] for first beat

Module Instance	Base Address	Register Address
ecc_hmc_ocp_slv_block	0xFFCFB000	0xFFCFB010

Offset: 0x10

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MPR0 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MPR0 0x0															

MPR_0BEAT1 Fields

Bit	Name	Description	Access	Reset
31:0	MPR0	MPR reg[31:0] for first beat	RO	0x0

MPR_1BEAT1

MPR register [63:32] for first beat

Module Instance	Base Address	Register Address
ecc_hmc_ocp_slv_block	0xFFCFB000	0xFFCFB014

Offset: 0x14

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MPR32 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MPR32 0x0															

MPR_1BEAT1 Fields

Bit	Name	Description	Access	Reset
31:0	MPR32	MPR reg[63:32] for first beat	RO	0x0

MPR_2BEAT1

MPR register [95:64] for first beat

Module Instance	Base Address	Register Address
ecc_hmc_ocp_slv_block	0xFFCFB000	0xFFCFB018

Offset: 0x18

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MPR64 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MPR64 0x0															

MPR_2BEAT1 Fields

Bit	Name	Description	Access	Reset
31:0	MPR64	MPR reg[95:64] for first beat	RO	0x0

MPR_3BEAT1

MPR register [127:96] for first beat

Module Instance	Base Address	Register Address
ecc_hmc_ocp_slv_block	0xFFCFB000	0xFFCFB01C

Offset: 0x1C

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MPR96 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MPR96 0x0															

MPR_3BEAT1 Fields

Bit	Name	Description	Access	Reset
31:0	MPR96	MPR reg[127:96] for first beat	RO	0x0

MPR_4BEAT1

MPR register [159:128] for first beat

Module Instance	Base Address	Register Address
ecc_hmc_ocp_slv_block	0xFFCFB000	0xFFCFB020

Offset: 0x20

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MPR128 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MPR128 0x0															

MPR_4BEAT1 Fields

Bit	Name	Description	Access	Reset
31:0	MPR128	MPR reg[159:128] for first beat	RO	0x0

MPR_5BEAT1

MPR register [191:160] for first beat

Module Instance	Base Address	Register Address
ecc_hmc_ocp_slv_block	0xFFCFB000	0xFFCFB024

Offset: 0x24

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MPR160 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MPR160 0x0															

MPR_5BEAT1 Fields

Bit	Name	Description	Access	Reset
31:0	MPR160	MPR reg[191:160] for first beat	RO	0x0

MPR_6BEAT1

MPR register [223:192] for first beat

Module Instance	Base Address	Register Address
ecc_hmc_ocp_slv_block	0xFFCFB000	0xFFCFB028

Offset: 0x28

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MPR192 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MPR192 0x0															

MPR_6BEAT1 Fields

Bit	Name	Description	Access	Reset
31:0	MPR192	MPR reg[223:192] for first beat	RO	0x0

MPR_7BEAT1

MPR register [255:224] for first beat

Module Instance	Base Address	Register Address
ecc_hmc_ocp_slv_block	0xFFCFB000	0xFFCFB02C

Offset: 0x2C

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MPR224 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MPR224 0x0															

MPR_7BEAT1 Fields

Bit	Name	Description	Access	Reset
31:0	MPR224	MPR reg[255:224] for first beat	RO	0x0

MPR_8BEAT1

MPR register [287:256] for first beat

Module Instance	Base Address	Register Address
ecc_hmc_ocp_slv_block	0xFFCFB000	0xFFCFB030

Offset: 0x30

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MPR256 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MPR256 0x0															

MPR_8BEAT1 Fields

Bit	Name	Description	Access	Reset
31:0	MPR256	MPR reg[287:256] for first beat	RO	0x0

MPR_0BEAT2

MPR register [31:0] for second beat

Module Instance	Base Address	Register Address
ecc_hmc_ocp_slv_block	0xFFCFB000	0xFFCFB034

Offset: 0x34

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MPR0 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MPR0 0x0															

MPR_0BEAT2 Fields

Bit	Name	Description	Access	Reset
31:0	MPR0	MPR reg[31:0] for second beat	RO	0x0

MPR_1BEAT2

MPR register [63:32] for second beat

Module Instance	Base Address	Register Address
ecc_hmc_ocp_slv_block	0xFFCFB000	0xFFCFB038

Offset: 0x38

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MPR32 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MPR32 0x0															

MPR_1BEAT2 Fields

Bit	Name	Description	Access	Reset
31:0	MPR32	MPR reg[63:32] for second beat	RO	0x0

MPR_2BEAT2

MPR register [95:64] for second beat

Module Instance	Base Address	Register Address
ecc_hmc_ocp_slv_block	0xFFCFB000	0xFFCFB03C

Offset: 0x3C

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MPR64 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MPR64 0x0															

MPR_2BEAT2 Fields

Bit	Name	Description	Access	Reset
31:0	MPR64	MPR reg[95:64] for second beat	RO	0x0

MPR_3BEAT2

MPR register [127:96] for second beat

Module Instance	Base Address	Register Address
ecc_hmc_ocp_slv_block	0xFFCFB000	0xFFCFB040

Offset: 0x40

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MPR96 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MPR96 0x0															

MPR_3BEAT2 Fields

Bit	Name	Description	Access	Reset
31:0	MPR96	MPR reg[127:96] for second beat	RO	0x0

MPR_4BEAT2

MPR register [159:128] for second beat

Module Instance	Base Address	Register Address
ecc_hmc_ocp_slv_block	0xFFCFB000	0xFFCFB044

Offset: 0x44

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MPR128 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MPR128 0x0															

MPR_4BEAT2 Fields

Bit	Name	Description	Access	Reset
31:0	MPR128	MPR reg[159:128] for second beat	RO	0x0

MPR_5BEAT2

MPR register [191:160] for second beat

Module Instance	Base Address	Register Address
ecc_hmc_ocp_slv_block	0xFFCFB000	0xFFCFB048

Offset: 0x48

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MPR160 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MPR160 0x0															

MPR_5BEAT2 Fields

Bit	Name	Description	Access	Reset
31:0	MPR160	MPR reg[191:160] for second beat	RO	0x0

MPR_6BEAT2

MPR register [223:192] for second beat

Module Instance	Base Address	Register Address
ecc_hmc_ocp_slv_block	0xFFCFB000	0xFFCFB04C

Offset: 0x4C

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MPR192 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MPR192 0x0															

MPR_6BEAT2 Fields

Bit	Name	Description	Access	Reset
31:0	MPR192	MPR reg[223:192] for second beat	RO	0x0

MPR_7BEAT2

MPR register [255:224] for second beat

Module Instance	Base Address	Register Address
ecc_hmc_ocp_slv_block	0xFFCFB000	0xFFCFB050

Offset: 0x50

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MPR224 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MPR224 0x0															

MPR_7BEAT2 Fields

Bit	Name	Description	Access	Reset
31:0	MPR224	MPR reg[255:224] for second beat	RO	0x0

MPR_8BEAT2

MPR register [287:256] for second beat

Module Instance	Base Address	Register Address
ecc_hmc_ocp_slv_block	0xFFCFB000	0xFFCFB054

Offset: 0x54

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MPR256 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MPR256 0x0															

MPR_8BEAT2 Fields

Bit	Name	Description	Access	Reset
31:0	MPR256	MPR reg[287:256] for second beat	RO	0x0

AUTO_PRECHARGE

Auto-precharge register

Module Instance	Base Address	Register Address
ecc_hmc_ocp_slv_block	0xFFCFB000	0xFFCFB060

Offset: 0x60

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															CTRL 0x0

AUTO_PRECHARGE Fields

Bit	Name	Description	Access	Reset
0	CTRL	Control bit to drive auto-precharge bit on command interface to the hard memory controller. 1'b0:Default value after reset. Auto-precharge request to the hard memory controller is disabled 1'b1: Every read/write command sent to the hard memory controller has the auto-precharge bit enabled	RW	0x0

ECCCTRL1

ECC control 1.
This bit is used to set the initialize the memory and ecc to a known value

Module Instance	Base Address	Register Address
ecc_hmc_ocp_slv_block	0xFFCFB000	0xFFCFB100

Offset: 0x100

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															AUTOWB_CNT_RST 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							CNT_RST 0x0	Reserved							ECC_EN 0x0

ECCCTRL1 Fields

Bit	Name	Description	Access	Reset
16	AUTOWB_CNT_RST	Reset the auto-write back internal counter to zero. The auto-write back counter is reset to zero when the INTONCMP bit in the INTMODE register is set. 1'b0 : No effect on auto-write back internal counter. Default value after reset 1'b1 : Reset the auto-write back internal counter to zero	RW	0x0
8	CNT_RST	Reset of internal counter. The internal counter is reset to zero when the INTONCMP bit in the INTMODE register is set. 1'b0: No effect on internal counter. Dafault value after reset 1'b1: Reset the internal counter to zero	RW	0x0
0	ECC_EN	Enable for the ECC detection and correction logic. This bit is used to detect and correct any errors within RAM during memory access. 1'b0:ECC block is disabled. Default value after reset. 1'b1: ECC block is enabled. Every RAM access will verify the data and generate any necessary error requests.	RW	0x0

ECCCTRL2

ECC control 2.

This bit is used to set the initialize the memory and ecc to a known value

Module Instance	Base Address	Register Address
ecc_hmc_ocp_slv_block	0xFFCFB000	0xFFCFB104

Offset: 0x104

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															OVRW_RB_ECC_EN 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							RMW_EN 0x0	Reserved							AUTOWB_EN 0x0

ECCCTRL2 Fields

Bit	Name	Description	Access	Reset
16	OVRW_RB_ECC_EN	Overwrite the read-back ecc code during RMW process if a double byte error is detected. 1'b0: write the read-back ECC from RMW process if derr is detected. Default value after reset. 1'b1: write of 1 will overwrite the ECC overwrite feature.	RW	0x0
8	RMW_EN	Enable read modify write logic. When ECC is enabled and sub word accesses require correct ECC to be calculated, this bit should be enabled. RMW_EN bit should be disabled when ECC_EN is disabled. 1'b0: disable RMW logic. Default value after reset. 1'b1: enable RMW logic.	RW	0x0

Bit	Name	Description	Access	Reset
0	AUTOWB_EN	Enable auto write back correction feature. When a single-bit error is detected on outgoing reads, the hard memory controller adaptor schedules the corrected data and ECC to the written to the DDR memory. This bit enables auto correction of DDR memory. 1'b0: disable auto WB drop correction. Default value after reset. 1'b1: enable auto WB drop correction.	RW	0x0

ERRINTEN

Error Interrupt enable

Module Instance	Base Address	Register Address
ecc_hmc_ocp_slv_block	0xFFCFB000	0xFFCFB110

Offset: 0x110

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													HMI_INTREN 0x0	DERRINTEN 0x0	SERRINTEN 0x0

ERRINTEN Fields

Bit	Name	Description	Access	Reset
2	HMI_INTREN	Enables GP HMI interrupt. This bit is used to enable the general purpose HMI interrupt error interrupt to system manager. When this bit is enabled along with autoWB_drop_en, it compares the internal counter with autoWB_drop_cntreg value. If the value is greater than or equal to, then the interrupt will be asserted.. 1'b0: hmi interrupt generation logic is disabled. 1'b1: hmi interrupt generation logic is enabled.	RW	0x0
1	DERRINTEN	This bit is used to enable the double bit error interrupt to system manager.When dbe error occurs, bus error is always generated with the transaction.DERR interrupt (derr_req)will be generated when this bit is enabled. 1'b0: DBE interrupt generation logic is disabled. 1'b1: DBE interrupt generation logic is enabled,	RW	0x0
0	SERRINTEN	This bit is used to enable the single bit error to system manager. It enables the interrupt modes (sbe request,compare match) 1'b0: SBE interrupt generation logic is disabled. 1'b1: SBE interrupt generation logic is enabled,	RW	0x0

ERRINTENS

Error Interrupt set

Module Instance	Base Address	Register Address
ecc_hmc_ocp_slv_block	0xFFCFB000	0xFFCFB114

Offset: 0x114

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													HMI_INTRS 0x0	DERRINTS 0x0	SERRINTS 0x0

ERRINTENS Fields

Bit	Name	Description	Access	Reset
2	HMI_INTRS	This bit is used to set the general purposes HMI interrupt error. 1'b0: writing of zero has no effect 1'b1: writing one, HMI_INTREN bit to 1. This is performing a bitwise writing of this feature.	RW	0x0
1	DERRINTS	This bit is used to set the double-bit error interrupt bit. Reads reflect DERRINTEN. 1'b0: writing of zero has no effect 1'b1: writing one, DERRINTEN bit to 1. This is performing a bitwise writing of this feature.	RW	0x0
0	SERRINTS	This bit is used to set the single-bit error interrupt bit. Reads reflect SERRINTEN. 1'b0: writing of zero has no effect 1'b1: writing one, this bit will set SERRINTEN bit to 1. This is performing a bitwise writing of this feature.	RW	0x0

ERRINTENR

Error Interrupt reset.

Module Instance	Base Address	Register Address
ecc_hmc_ocp_slv_block	0xFFCFB000	0xFFCFB118

Offset: 0x118

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													HMI_INTRR 0x0	DERRINTR 0x0	SERRINTR 0x0

ERRINTENR Fields

Bit	Name	Description	Access	Reset
2	HMI_INTRR	This bit is used to reset the general purpose HMI interrupt error interrupt to system manager 1'b0: Writing of zero has no effect. 1'b1: By writing one, this bit will reset HMI_INTREN bit to 0. This is performing a bitwise writing of this feature.	RW	0x0
1	DERRINTR	This bit is used to reset the double-bit error interrupt bit. Reads reflect DERRINTEN. 1'b0: Writing of zero has no effect. 1'b1: By writing one, this bit will reset DERRINTEN bit to 0. This is performing a bitwise writing of this feature.	RW	0x0
0	SERRINTR	This bit is used to reset the single-bit error interrupt bit. Reads reflect SERRINTEN. 1'b0: Writing of zero has no effect. 1'b1: By writing one, this bit will reset SERRINTEN bit to 0. This is performing a bitwise writing of this feature.	RW	0x0

INTMODE

Interrupt mode

Module Instance	Base Address	Register Address
ecc_hmc_ocp_slv_block	0xFFCFB000	0xFFCFB11C

Offset: 0x11C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							AFICAL_EN 0x0	Reserved							INTONCMP 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							EXT_ADDRPARITY_EN 0x0	Reserved							INTMODE 0x0

INTMODE Fields

Bit	Name	Description	Access	Reset
24	AFICAL_EN	Enable interrupt of memory interface calibration success. This bit is used to enable interrupt of memory interrupt calibration success. The hmi_intr signal will be asserted on a match. 1'b0: HMI interrupts on compare match is disabled. 1'b1: HMI interrupts on compare matched is enabled.	RW	0x0

Bit	Name	Description	Access	Reset
16	INTONCMP	Enable interrupt on compare match. This bit is used to enable interrupt when the internal counter and SERRCNTA value matches. serr_req signal will be asserted on a match. 1'b0: SERR interrupt on compare match is disabled 1'b1: SERR interrupt on compare match is enabled	RW	0x0
8	EXT_ADDRPARITY_EN	Enable address parity for DDR4 memories. This bit is used to enable the interrupt that generate externally when address parity is detected. when enabled, this will be generating derr_req signal 1'b0: disable address parity on DERR interrupt 1'b1: enable address parity on DERR interrupt	RW	0x0
0	INTMODE	Interrupt mode for single-bit error.This is disabled when SERRINTEN is disabled in the ERRINTEN register. 1'b0: interrupt disbaled 1'b1: generate interrupt on every SERR	RW	0x0

INTSTAT

Interrupt status

Module Instance	Base Address	Register Address
ecc_hmc_ocp_slv_block	0xFFCFB000	0xFFCFB120

Offset: 0x120

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													DERRBUSFLG 0x0	ADDRPARFLG 0x0	ADDRMTCFLG 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													HMI_PENA 0x0	DERRPENA 0x0	SERRPENA 0x0

INTSTAT Fields

Bit	Name	Description	Access	Reset
18	DERRBUSFLG	This bit is used to indicate that the last transaction was flagged with double-bit error. 1'b0: no effect. 1'b1: indicates double-bit error has occured. This will drive the bus to respond the read with bus error. Write of one will clear this register double-but bus error. Bus error occurs as part of the transaction but this indicates the SW the cause of the error. This should only occur once per transaction	RW	0x0
17	ADDRPARFLG	External address parity flag for DDR4 memory. This bit is used to flag external address parity flag which is driven with derr_req port. 1'b0: No Effect. 1'b1: Read of one indicates double-bit interrupt has occurred. Write of one will clear this register last address parity flag.	RW	0x0

Bit	Name	Description	Access	Reset
16	ADDRMTCFLG	Address mismatch error flag. This bit is used to indicate the last transaction was flagged with address mismatch error. 1'b0: No effect. 1'b1: indicates address mismatch error has occured. This will drive the bus to respond the read with bus error. Write of one will clears this register address mismatch error. Bus error occurs as part of the transaction but this indicates the SW the cause of the error. This should occur once per transaction.	RW	0x0
2	HMI_PENA	HMI interrupt pending This bit is used to clear the pending hmi interrupt bit. 1'b0: No effect 1'b1: indicates hmi interrupt is pending. Write of one will clear the pending interrupt. This will de-assert the hmi_intr signal.	RW	0x0
1	DERRPENA	Double bit error pending This bit is used to clear the pending double-bit error. 1'b0: No effect. 1'b1: indicates DBE is pending. Write of one will clear the pending DBE. This will de-assert the derr_req signal.	RW	0x0
0	SERRPENA	Single-bit error pending This bit is used to clear the pending single-bit error. 1'b0: No effect. 1'b1: indicates SBE is pending. Write of one will clear the pending. This will de-assert the serr_req signal.	RW	0x0

DIAGINTTEST

Enable diagnostic errors

Module Instance	Base Address	Register Address
ecc_hmc_ocp_slv_block	0xFFCFB000	0xFFCFB124

Offset: 0x124

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							TADDRPAR 0x0	Reserved							TADDRMTC 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							TDERRA 0x0	Reserved							TSERRA 0x0

DIAGINTTEST Fields

Bit	Name	Description	Access	Reset
24	TADDRPAR	Diagnostic of address parity of DDR4. This bit is used to test the address parity error path. 1'b0: Disables generating address match bus error as part of the transaction. 1'b1: When this bit is set to 1, derr_req signal is generated to the system manager. By writing to this bit, ADDRPARFLG bit in the INTSTAT register will be pending. Write of one to ADDRPARFLG will clear this bit. SW needs to explicitly write to DERRPENA to clear it.	RW	0x0

Bit	Name	Description	Access	Reset
16	TADDRMTC	Diagnostic enable of Address mismatch error. This bit is used to flag that the last transaction was flagged with address mismatch error. 1'b0: Disables generating address match bus error as part of the transaction. 1'b1: When this bit is set to 1, derr_req signal is generated to the system manager without going through the ECC decoder address mismatch logic. By writing to this bit, ADDRMTCFLG bit in the INTSTAT register will be pending. Write of one to ADDRMTCFLG will clear this bit. SW needs to explicitly write to DERRPENA to clear it.	RW	0x0
8	TDERRA	Diagnostic enable of Double-bit error. This bit is used to test double-bit error. 1'b0: Write of zero has no effect. 1'b1: When this bit is set to 1, derr_req signal is generated to the system manager without going through the ECC decoder logic. By writing to this bit, DERRBUSFLG bit in the INTSTAT register will be pending. Write of one to DERRBUSFLG will clear this bit. SW needs to explicitly write to DERRPENA to clear it.	RW	0x0
0	TSERRA	This bit is used to test a single-bit error. 1'b0: Write of zero has no effect. 1'b1: When this bit is set to 1, serr_req signal is generated to the system manager without going through the ECC decoder logic. By writing to this bit, SERRPENA bit in the INTSTAT register will be pending. Write of one to SERRPENA will clear this bit.	RW	0x0

MODSTAT

Counter feature status flag

Module Instance	Base Address	Register Address
ecc_hmc_ocp_slv_block	0xFFCFB000	0xFFCFB128

Offset: 0x128

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							AUTOWB_DROP_FLG 0x0	Reserved							CMPFLGA 0x0

MODSTAT Fields

Bit	Name	Description	Access	Reset
8	AUTOWB_DROP_FLG	Auto writeback counter match flag. This bit indicates that the internal autoWB counter and autoWB_drop_cnt value matched. 1'b0: read indicates match check of hmi_intr interrupt on compare match is disabled. 1'b1: read indicates compare has matched. Write of one will clear the pending compare match. This will not de-assert the hmi_intr signal - software needs to write to hmi_intrpen bit to clear the interrupt. When the match occurs, additional errors will not increment count until the compare status flag is cleared. If the software does not change the autoWB_drop_cnt register prior to clearing this flag or reset the autoWB counter, next increment of internal autoWB counter could set this flag in the next cycle.	RW	0x0
0	CMPFLGA	Counter Match occurred flag. This bit indicates that the internal counter and SERRCNT value matched. 1'b0: read indicates match check of SERR interrupt on compare match is disabled. 1'b1: read indicates compare has matched. Write of one will clear the pending compare match. This will not de-assert the serr_req signal - software needs to write to serrpen bit to clear the interrupt. When the match occurs, additional errors will not increment count until the compare status flag is cleared. If the software does not change the SERRCNT register prior to clearing this flag or reset the internal counter, next increment of internal counter could set this flag again in the next cycle.	RW	0x0

DERRADDRA

Double-bit error address

Module Instance	Base Address	Register Address
ecc_hmc_ocp_slv_block	0xFFCFB000	0xFFCFB12C

Offset: 0x12C

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DADDRESS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DADDRESS 0x0															

DERRADDRA Fields

Bit	Name	Description	Access	Reset
31:0	DADDRESS	Last double-bit error address at 256-bit boundary. This register shows the address of the current double-bit error. RAM size will determine the maximum number of address bits. This address is logged when a new derr_req or bus error is generated to the system. This is gated by the ecc_en enable bit and derrinten bit.	RO	0x0

SERRADDRA

Single-bit error address

Module Instance	Base Address	Register Address
ecc_hmc_ocp_slv_block	0xFFCFB000	0xFFCFB130

Offset: 0x130

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SADDRESS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SADDRESS 0x0															

SERRADDRA Fields

Bit	Name	Description	Access	Reset
31:0	SADDRESS	Last single-bit error address at 256-bit boundary. This register shows the address of the current single-bit error. This address is logged when a new serr_req is generated to the system. This is gated by the single-bit error interrupt enable and ecc_en.	RO	0x0

AUTOWB_CORRADDR

This register shows the address of the current autoWB correction SBE.

Module Instance	Base Address	Register Address
ecc_hmc_ocp_slv_block	0xFFCFB000	0xFFCFB138

Offset: 0x138

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SWBADDRESS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SWBADDRESS 0x0															

AUTOWB_CORRADDR Fields

Bit	Name	Description	Access	Reset
31:0	SWBADDRESS	recent autoWB correction address. This register shows the address of the current autoWB correction single-bit error. This address is logged when a new serr_req is generated to the system. This is gated by the single-bit error interrupt enable and ecc_en. Note that when a single-bit error occurs, an address is logged regardless of whether auto-correction is enabled or not. If you have not enabled auto-correction, this register can be ignored.	RO	0x0

SERRCNTREG

Maximum counter value for single-bit error interrupt

Module Instance	Base Address	Register Address
ecc_hmc_ocp_slv_block	0xFFCFB000	0xFFCFB13C

Offset: 0x13C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SERRCNT 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SERRCNT 0x0															

SERRCNTREG Fields

Bit	Name	Description	Access	Reset
31:0	SERRCNT	Compare value for the internal single-bit errors. This register sets the value to compare with the internal counter. Software should write to this register before enabling the interrupt on compare. 0x0: If the serrcnt bits are not modified before enabling the intoncmp, internal counter=0 and serrcnt=0, serr compare interrupt will not occur. Default after reset. Nonzero: if internal counter == serrcnt == nonzero will create a serr compare interrupt. When the compare matches, autoWB_drop_cmpflga will be set.	RW	0x0

AUTOWB_DROP_CNTREG

Maximum counter value for AUTOWB correction interrupt

Module Instance	Base Address	Register Address
ecc_hmc_ocp_slv_block	0xFFCFB000	0xFFCFB140

Offset: 0x140

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CNT 0x1															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT 0x1															

AUTOWB_DROP_CNTREG Fields

Bit	Name	Description	Access	Reset
31:0	CNT	Compare value for the internal autoWB correction count. This register sets the value to compare with the autoWB internal counter. Software should write to this register before enabling the interrupt on compare. 0x1: If the autoWB_drop_cntreg bits are not modified before enabling the hmi_intr, autoWB internal counter=0 and autoWB_dop_cnt =1, serr compare interrupt will not occur. Default after reset. Nonzero: if autoWB internal counter == autoWB_drop_cnt == nonzero will create a serr compare interrupt. When the compare matches, autoWB_ drop_flg will be set.	RW	0x1

ECC_REG2WRECCDATABUS

ECC from register associated to data which will be written to the RAM

Module Instance	Base Address	Register Address
ecc_hmc_ocp_slv_block	0xFFCFB000	0xFFCFB144

Offset: 0x144

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC3BUS 0x0								ECC2BUS 0x0							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC1BUS 0x0								ECC0BUS 0x0							

ECC_REG2WRECCDATABUS Fields

Bit	Name	Description	Access	Reset
31:24	ECC3BUS	ECC from register associated to data [255:192] which will be written to the RAM	RW	0x0
23:16	ECC2BUS	ECC from register associated to data [191:128] which will be written to the RAM	RW	0x0
15:8	ECC1BUS	ECC from register associated to data [127:64] which will be written to the RAM	RW	0x0
7:0	ECC0BUS	ECC from register associated to data [63:0] which will be written to the RAM	RW	0x0

ECC_REG2RDECCDATABUS

ECC from register associated to RD data which will be written to hmc ecc

Module Instance	Base Address	Register Address
ecc_hmc_ocp_slv_block	0xFFCFB000	0xFFCFB14C

Offset: 0x14C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC3BUS 0x0								ECC2BUS 0x0							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC1BUS 0x0								ECC0BUS 0x0							

ECC_REG2RDECCDATABUS Fields

Bit	Name	Description	Access	Reset
31:24	ECC3BUS	ECC from register associated to RD data [255:192] which will be written to hmc ecc. Based on the DDR IO width, unimplemented bytes of this register will read as zero.	RW	0x0
23:16	ECC2BUS	ECC from register associated to RD data [191:128] which will be written to hmc ecc. Based on the DDR IO width, unimplemented bytes of this register will read as zero.	RW	0x0
15:8	ECC1BUS	ECC from register associated to RD data [127:64] which will be written to hmc ecc. Based on the DDR IO width, unimplemented bytes of this register will read as zero.	RW	0x0
7:0	ECC0BUS	ECC from register associated to RD data [63:0] which will be written to hmc ecc. Based on the DDR IO width, unimplemented bytes of this register will read as zero.	RW	0x0

ECC_DIAGON

Enable diagnostics access

Module Instance	Base Address	Register Address
ecc_hmc_ocp_slv_block	0xFFCFB000	0xFFCFB150

Offset: 0x150

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															ECCDIAGON 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														RDDIAGON 0x0	WRDIAGON 0x0

ECC_DIAGON Fields

Bit	Name	Description	Access	Reset
16	ECCDIAGON	ECC diagnostics mode. 1'b0: ECC diagnostics logic is disabled. ECC encoder bypass is disabled. 1'b1: ECC diagnostics logic is enabled. Direction of ECC data from the register to data bus or data bus to ecc register is determined by ECC_rddiagon or ECC_wrdiagon.	RW	0x0
1	RDDIAGON	Read diagnostics mux enabled. This overrides the data entering the ECC decoder. 1'b0: Read diagnostics path via the ecc_rdata2regbus or ecc_reg2rdatabus is disabled. 1'b1: Read diagnostics path via the ecc_rdata2regbus or ecc_reg2rdatabus is enabled. Both Rddiagon and Wrdiagon bits can be enabled.	RW	0x0

Bit	Name	Description	Access	Reset
0	WRDIAGON	Write diagnostics mux enabled. This overrides the encoder output with the register data ecc. 1'b0: Write diagnostics path via the ecc_reg2wdatabus is disabled. 1'b1: Write diagnostics path via the ecc_reg2wdatabus is enabled. Both Rddiagon and Wrdiagon bits can be enabled.	RW	0x0

ECC_DECSTAT

Diagnostic decoder status

Module Instance	Base Address	Register Address
ecc_hmc_ocp_slv_block	0xFFCFB000	0xFFCFB154

Offset: 0x154

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				DEC3DERRFLG 0x0	DEC2DERRFLG 0x0	DEC1DERRFLG 0x0	DEC0DERRFLG 0x0	DEC3ADDRFLG 0x0	DEC2ADDRFLG 0x0	DEC1ADDRFLG 0x0	DEC0ADDRFLG 0x0	DEC3SERRFLG 0x0	DEC2SERRFLG 0x0	DEC1SERRFLG 0x0	DEC0SERRFLG 0x0

ECC_DECSTAT Fields

Bit	Name	Description	Access	Reset
11	DEC3DERRFLG	indicates decoder for data [255:192] has detected DBE. 1'b0: No error has been captured with this flag. 1'b1: Decoder 0 detected a double-bit error. This flag will be set by the hardware and it will be cleared by the writing 1. This flag will be set till SW clears. Additional errors will not change the state of this bit. Error flag is set on the first beat of erred data. This won't be reset by the ecc_en bit.	RW	0x0
10	DEC2DERRFLG	indicates decoder for data [191:128] has detected DBE. 1'b0: No error has been captured with this flag. 1'b1: Decoder 0 detected a double-bit error. This flag will be set by the hardware and it will be cleared by the writing 1. This flag will be set till SW clears. Additional errors will not change the state of this bit.Error flag is set on the first beat of erred data. This won't be reset by the ecc_en bit.	RW	0x0
9	DEC1DERRFLG	indicates decoder for data [127:64] has detected DBE. 1'b0: No error has been captured with this flag. 1'b1: Decoder 0 detected a double-bit error. This flag will be set by the hardware and it will be cleared by the writing 1. This flag will be set till SW clears. Additional errors will not change the state of this bit.Error flag is set on the first beat of erred data. This won't be reset by the ecc_en bit.	RW	0x0

Bit	Name	Description	Access	Reset
8	DEC0DERRFLG	indicates decoder for data [63:0] has detected DBE. 1'b0: No error has been captured with this flag. 1'b1: Decoder 0 detected a double-bit error. This flag will be set by the hardware and it will be cleared by the writing 1. This flag will be set till SW clears. Additional errors will not change the state of this bit. Error flag is set on the first beat of erred data. This won't be reset by the ecc_en bit.	RW	0x0
7	DEC3ADDRFLG	indicates decoder for data [255:192] has detected address error. 1'b0: No error has been captured with this flag. 1'b1: Decoder 0 detected an address mismatch error. This flag will be set by the hardware and it will be cleared by the writing 1. This flag will be set till SW clears.Additional errors will not change the state of this bit.Error flag is set on the first beat of erred data. This won't be reset by the ecc_en bit.	RW	0x0
6	DEC2ADDRFLG	indicates decoder for data [191:128] has detected address error. 1'b0: No error has been captured with this flag. 1'b1: Decoder 0 detected an address mismatch error. This flag will be set by the hardware and it will be cleared by the writing 1. This flag will be set till SW clears.Additional errors will not change the state of this bit.Error flag is set on the first beat of erred data. This won't be reset by the ecc_en bit.	RW	0x0

Bit	Name	Description	Access	Reset
5	DEC1ADDRFLG	indicates decoder for data [127:64] has detected address error. 1'b0: No error has been captured with this flag. 1'b1: Decoder 0 detected an address mismatch error. This flag will be set by the hardware and it will be cleared by the writing 1. This flag will be set till SW clears.Additional errors will not change the state of this bit.Error flag is set on the first beat of erred data. This won't be reset by the ecc_en bit.	RW	0x0
4	DEC0ADDRFLG	indicates decoder for data [63:0] has detected address error. 1'b0: No error has been captured with this flag. 1'b1: Decoder 0 detected an address mismatch error. This flag will be set by the hardware and it will be cleared by the writing 1. This flag will be set till SW clears.Additional errors will not change the state of this bit.Error flag is set on the first beat of erred data. This won't be reset by the ecc_en bit.	RW	0x0
3	DEC3SERRFLG	indicates decoder for data [255:192] has detected SBE. 1'b0: No error has been captured with this flag. 1'b1: Decoder 0 detected a single bit error. This flag will be set by the hardware and it will be cleared by the writing 1. This flag will be set till SW clears. Additional errors will not change the state of this bit.Error flag is set on the first beat of erred data. This won't be reset by the ecc_en bit.	RW	0x0

Bit	Name	Description	Access	Reset
2	DEC2SERRFLG	indicates decoder for data [191:128] has detected SBE. 1'b0: No error has been captured with this flag. 1'b1: Decoder 0 detected a single bit error. This flag will be set by the hardware and it will be cleared by the writing 1. This flag will be set till SW clears. Additional errors will not change the state of this bit.Error flag is set on the first beat of erred data. This won't be reset by the ecc_en bit.	RW	0x0
1	DEC1SERRFLG	indicates decoder for data [127:64] has detected SBE. 1'b0: No error has been captured with this flag. 1'b1: Decoder 0 detected a single bit error. This flag will be set by the hardware and it will be cleared by the writing 1. This flag will be set till SW clears. Additional errors will not change the state of this bit. Error flag is set on the first beat of erred data. This won't be reset by the ecc_en bit.	RW	0x0
0	DEC0SERRFLG	indicates decoder for data [63:0] has detected SBE. 1'b0: No error has been captured with this flag. 1'b1: Decoder 0 detected a single bit error. This flag will be set by the hardware and it will be cleared by the writing 1. This flag will be set till SW clears. Additional errors will not change the state of this bit. Error flag is set on the first beat of erred data. This won't be reset by the ecc_en bit.	RW	0x0

ECC_ERRGENADDR_0

Error address register

Module Instance	Base Address	Register Address
ecc_hmc_ocp_slv_block	0xFFCFB000	0xFFCFB160

Offset: 0x160

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDR 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR 0x0															

ECC_ERRGENADDR_0 Fields

Bit	Name	Description	Access	Reset
31:0	ADDR	For decoder 0. Address generated with SER or address mismatch logic. Address will be driven by the ECC decoder on every read latched by the RAM independent of ECCDiagon is on.	RO	0x0

ECC_ERRGENADDR_1

Error address register

Module Instance	Base Address	Register Address
ecc_hmc_ocp_slv_block	0xFFCFB000	0xFFCFB164

Offset: 0x164

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDR 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR 0x0															

ECC_ERRGENADDR_1 Fields

Bit	Name	Description	Access	Reset
31:0	ADDR	For decoder 1. Address generated with SER or address mismatch logic. Address will be driven by the ECC decoder on every read latched by the RAM independent of ECCDiagon is on.	RO	0x0

ECC_ERRGENADDR_2

Error address register

Module Instance	Base Address	Register Address
ecc_hmc_ocp_slv_block	0xFFCFB000	0xFFCFB168

Offset: 0x168

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDR 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR 0x0															

ECC_ERRGENADDR_2 Fields

Bit	Name	Description	Access	Reset
31:0	ADDR	For decoder 2. Address generated with SER or address mismatch logic. Address will be driven by the ECC decoder on every read latched by the RAM independent of ECCDiagon is on.	RO	0x0

ECC_ERRGENADDR_3

Error address register

Module Instance	Base Address	Register Address
ecc_hmc_ocp_slv_block	0xFFCFB000	0xFFCFB16C

Offset: 0x16C

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDR 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR 0x0															

ECC_ERRGENADDR_3 Fields

Bit	Name	Description	Access	Reset
31:0	ADDR	For decoder 3. Address generated with SER or address mismatch logic. Address will be driven by the ECC decoder on every read latched by the RAM independent of ECCDiagon is on.	RO	0x0

ECC_REG2RDDATABUS_BEAT0

ECC Reg2Rddatabus_beat0

Module Instance	Base Address	Register Address
ecc_hmc_ocp_slv_block	0xFFCFB000	0xFFCFB170

Offset: 0x170

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC3BUS 0x0								ECC2BUS 0x0							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC1BUS 0x0								ECC0BUS 0x0							

ECC_REG2RDDATABUS_BEAT0 Fields

Bit	Name	Description	Access	Reset
31:24	ECC3BUS	Data ECC from the register will be written to the RAM. Based on the DDR IO width, unimplemented bytes of this register will read as zero. Each (reg2rddatabus_beat[3:0]) register will be shifted in every beat cycle When data is 64 bit wide: Data ECC is associated with data[255:192]. When data is 32 bit wide: Data ECC is associated with data[31:0]. When data is 16 bit wide: Data ECC is associated with data[15:0].	RW	0x0

Bit	Name	Description	Access	Reset
23:16	ECC2BUS	Data ECC from the register will be written to the RAM. Based on the DDR IO width, unimplemented bytes of this register will read as zero. When data is 64 bit wide: Data ECC is associated with data[191:128]. When data is 32 bit wide: Data ECC is associated with data[159:128]. When data is 16 bit wide: Data ECC is associated with data[143:128].	RW	0x0
15:8	ECC1BUS	Data ECC from the register will be written to the RAM. Based on the DDR IO width, unimplemented bytes of this register will read as zero. When data is 64 bit wide: Data ECC is associated with data[127:64]. When data is 32 bit wide: Data ECC is associated with data[63:32]. When data is 16 bit wide: Data ECC is associated with data[31:16].	RW	0x0
7:0	ECC0BUS	Data ECC from the register will be written to the RAM. Based on the DDR IO width, unimplemented bytes of this register will read as zero. When data is 64 bit wide: Data ECC is associated with data[63:0]. When data is 32 bit wide: Data ECC is associated with data[31:0]. When data is 16 bit wide: Data ECC is associated with data[15:0].	RW	0x0

ECC_REG2RDDATABUS_BEAT1

ECC Reg2Rddatabus_beat1

Module Instance	Base Address	Register Address
ecc_hmc_ocp_slv_block	0xFFCFB000	0xFFCFB174

Offset: 0x174

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC3BUS 0x0								ECC2BUS 0x0							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC1BUS 0x0								ECC0BUS 0x0							

ECC_REG2RDDATABUS_BEAT1 Fields

Bit	Name	Description	Access	Reset
31:24	ECC3BUS	Data ECC from the register will be written to the RAM. Based on the DDR IO width, unimplemented bytes of this register will read as zero. Each(reg2rddatabus_beat[3:0]) register will be shifted in every beat cycle When data is 32 bit wide: Data ECC is associated with data[223:192]. When data is 16 bit wide: Data ECC is associated with data[207:192].	RW	0x0
23:16	ECC2BUS	Data ECC from the register will be written to the RAM. Based on the DDR IO width, unimplemented bytes of this register will read as zero. When data is 32 bit wide: Data ECC is associated with data[159:128]. When data is 16 bit wide: Data ECC is associated with data[143:128].	RW	0x0
15:8	ECC1BUS	Data ECC from the register will be written to the RAM. Based on the DDR IO width, unimplemented bytes of this register will read as zero. When data is 32 bit wide: Data ECC is associated with data[95:64]. When data is 16 bit wide: Data ECC is associated with data[79:64].	RW	0x0

Bit	Name	Description	Access	Reset
7:0	ECC0BUS	Data ECC from the register will be written to the RAM. Based on the DDR IO width, unimplemented bytes of this register will read as zero. When data is 32 bit wide: Data ECC is associated with data[63:32]. When data is 16 bit wide: Data ECC is associated with data[31:16].	RW	0x0

ECC_REG2RDDATABUS_BEAT2

ECC Reg2Rddatabus_beat2

Module Instance	Base Address	Register Address
ecc_hmc_ocp_slv_block	0xFFCFB000	0xFFCFB178

Offset: 0x178

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC3BUS 0x0								ECC2BUS 0x0							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC1BUS 0x0								ECC0BUS 0x0							

ECC_REG2RDDATABUS_BEAT2 Fields

Bit	Name	Description	Access	Reset
31:24	ECC3BUS	Data ECC from the register will be written to the RAM. Based on the DDR IO width, unimplemented bytes of this register will read as zero. Each (reg2rddatabus_beat[3:0]) register will be shifted in every beat cycle When data is 16 bit wide: Data ECC is associated with data[223:208].	RW	0x0

Bit	Name	Description	Access	Reset
23:16	ECC2BUS	Data ECC from the register will be written to the RAM. Based on the DDR IO width, unimplemented bytes of this register will read as zero. When data is 16 bit wide: Data ECC is associated with data[159:144].	RW	0x0
15:8	ECC1BUS	Data ECC from the register will be written to the RAM. Based on the DDR IO width, unimplemented bytes of this register will read as zero. When data is 16 bit wide: Data ECC is associated with data[96:80].	RW	0x0
7:0	ECC0BUS	Data ECC from the register will be written to the RAM Based on the DDR IO width, unimplemented bytes of this register will read as zero. When data is 16 bit wide: Data ECC is associated with data[47:32].	RW	0x0

ECC_REG2RDDATABUS_BEAT3

ECC Reg2Rddatabus_beat3

Module Instance	Base Address	Register Address
ecc_hmc_ocp_slv_block	0xFFCFB000	0xFFCFB17C

Offset: 0x17C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC3BUS 0x0								ECC2BUS 0x0							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC1BUS 0x0								ECC0BUS 0x0							

ECC_REG2RDDATABUS_BEAT3 Fields

Bit	Name	Description	Access	Reset
31:24	ECC3BUS	Data ECC from the register will be written to the RAM. Based on the DDR IO width, unimplemented bytes of this register will read as zero. Each (reg2rddatabus_beat[3:0]) register will be shifted in every beat cycle When data is 16 bit wide: Data ECC is associated with data[255:240].	RW	0x0
23:16	ECC2BUS	Data ECC from the register will be written to the RAM. Based on the DDR IO width, unimplemented bytes of this register will read as zero. When data is 16 bit wide: Data ECC is associated with data[191:176].	RW	0x0
15:8	ECC1BUS	Data ECC from the register will be written to the RAM. Based on the DDR IO width, unimplemented bytes of this register will read as zero. When data is 16 bit wide: Data ECC is associated with data[127:112].	RW	0x0
7:0	ECC0BUS	Data ECC from the register will be written to the RAM Based on the DDR IO width, unimplemented bytes of this register will read as zero. When data is 16 bit wide: Data ECC is associated with data[63:48].	RW	0x0

noc_l4_priv_l4_priv_filter Address Map

Module Instance	Base Address	End Address
noc_l4_priv_l4_priv_filter	0xFFD11000	0xFFD110FF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
l4_priv on page 7-191	0x0	32	RW	0x0	This register controls access to various Peripherals depending on the privilege setting. By default, all slaves will be assumed as Privileged. To allow non-Privileged access to a slave, the corresponding bit for the slave must be set. Once set, both Privilege and non-Privileged transac-tions are allowed to the Slave. Note that the privilege filter only checks for transaction Privilege level, transaction Security is left to Firewalls. Firewalls therefore may still block transaction to Peripherals depending on Security configura-tions.
l4_priv_set on page 7-196	0x4	32	WO	0x0	Sets Region Enable field when written with 1
l4_priv_clear on page 7-199	0x8	32	WO	0x0	Clears Region Enable field when written with 1

noc_l4_priv_l4_priv_filter Summary

Base Address: 0xFFD11000

Register Address Offset	Bit Fields
noc_l4_priv_l4_priv_filter	

Register Address Offset	Bit Fields															
l4_priv 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	soc2fpga RW 0x0	lwsoc2fpga RW 0x0	uart1 RW 0x0	uart0 RW 0x0	sp_timer1 RW 0x0	sp_timer0 RW 0x0	i2c4 RW 0x0	i2c3 RW 0x0	i2c2 RW 0x0	i2c1 RW 0x0	i2c0 RW 0x0	gpio2 RW 0x0	gpio1 RW 0x0	gpio0 RW 0x0	sdmmc RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	qspi RW 0x0	emac3 0x0	emac2 RW 0x0	emac1 RW 0x0	emac0 RW 0x0	spi_slave1 RW 0x0	spi_slave0 RW 0x0	spi_master1 RW 0x0	spi_master0 RW 0x0	dma_secure RW 0x0	dma_nonsecure RW 0x0	usb1_register RW 0x0	usb0_register RW 0x0	qspi_data RW 0x0	nand_data RW 0x0	nand_register RW 0x0
l4_priv_set 0x4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	soc2fpga WO 0x0	lwsoc2fpga WO 0x0	uart1 WO 0x0	uart0 WO 0x0	sp_timer1 WO 0x0	sp_timer0 WO 0x0	i2c4 WO 0x0	i2c3 WO 0x0	i2c2 WO 0x0	i2c1 WO 0x0	i2c0 WO 0x0	gpio2 WO 0x0	gpio1 WO 0x0	gpio0 WO 0x0	sdmmc WO 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	qspi WO 0x0	emac3 0x0	emac2 WO 0x0	emac1 WO 0x0	emac0 WO 0x0	spi_slave1 WO 0x0	spi_slave0 WO 0x0	spi_master1 WO 0x0	spi_master0 WO 0x0	dma_secure WO 0x0	dma_nonsecure WO 0x0	usb1_register WO 0x0	usb0_register WO 0x0	qspi_data WO 0x0	nand_data WO 0x0	nand_register WO 0x0
l4_priv_clear 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	soc2fpga WO 0x0	lwsoc2fpga WO 0x0	uart1 WO 0x0	uart0 WO 0x0	sp_timer1 WO 0x0	sp_timer0 WO 0x0	i2c4 WO 0x0	i2c3 WO 0x0	i2c2 WO 0x0	i2c1 WO 0x0	i2c0 WO 0x0	gpio2 WO 0x0	gpio1 WO 0x0	gpio0 WO 0x0	sdmmc WO 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	qspi WO 0x0	emac3 0x0	emac2 WO 0x0	emac1 WO 0x0	emac0 WO 0x0	spi_slave1 WO 0x0	spi_slave0 WO 0x0	spi_master1 WO 0x0	spi_master0 WO 0x0	dma_secure WO 0x0	dma_nonsecure WO 0x0	usb1_register WO 0x0	usb0_register WO 0x0	qspi_data WO 0x0	nand_data WO 0x0	nand_register WO 0x0

l4_priv

This register controls access to various Peripherals depending on the privilege setting. By default, all slaves will be assumed as Privileged. To allow non-Privileged access to a slave, the corresponding bit for the slave must be set. Once set, both Privilege and non-Privileged transactions are allowed to the Slave. Note that the privilege filter only checks for transaction Privilege level,

transaction Security is left to Firewalls. Firewalls therefore may still block transaction to Peripherals depending on Security configurations.

Module Instance	Base Address	Register Address
noc_l4_priv_l4_priv_filter	0xFFD11000	0xFFD11000

Offset: 0x0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	soc2fpga RW 0x0	lwsoc2fpga RW 0x0	uart1 RW 0x0	uart0 RW 0x0	sp_timer1 RW 0x0	sp_timer0 RW 0x0	i2c4 RW 0x0	i2c3 RW 0x0	i2c2 RW 0x0	i2c1 RW 0x0	i2c0 RW 0x0	gpio2 RW 0x0	gpio1 RW 0x0	gpio0 RW 0x0	sdmmc RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
qspi RW 0x0	emac3 0x0	emac2 RW 0x0	emac1 RW 0x0	emac0 RW 0x0	spi_slave1 RW 0x0	spi_slave0 RW 0x0	spi_master1 RW 0x0	spi_master0 RW 0x0	dma_secure RW 0x0	dma_nonsecure RW 0x0	usb1_register RW 0x0	usb0_register RW 0x0	qspi_data RW 0x0	nand_data RW 0x0	nand_register RW 0x0

l4_priv Fields

Bit	Name	Description	Access	Reset
30	soc2fpga	Privilege bit for SOC2FPGA. When 0, only privileged transactions are allowed to slave. When 1, both privileged and non-privileged transactions are allowed to slave	RW	0x0
29	lwsoc2fpga	Privilege bit for Lightweight SOC2FPGA. When 0, only privileged transactions are allowed to slave. When 1, both privileged and non-privileged transactions are allowed to slave	RW	0x0

Bit	Name	Description	Access	Reset
28	uart1	Privilege bit for uart1. When 0, only privileged transactions are allowed to slave. When 1, both privileged and non-privileged transactions are allowed to slave	RW	0x0
27	uart0	Privilege bit for uart0. When 0, only privileged transactions are allowed to slave. When 1, both privileged and non-privileged transactions are allowed to slave	RW	0x0
26	sp_timer1	Privilege bit for sp_timer1. When 0, only privileged transactions are allowed to slave. When 1, both privileged and non-privileged transactions are allowed to slave	RW	0x0
25	sp_timer0	Privilege bit for sp_timer0. When 0, only privileged transactions are allowed to slave. When 1, both privileged and non-privileged transactions are allowed to slave	RW	0x0
24	i2c4	Privilege bit for i2c4. When 0, only privileged transactions are allowed to slave. When 1, both privileged and non-privileged transactions are allowed to slave	RW	0x0
23	i2c3	Privilege bit for i2c3. When 0, only privileged transactions are allowed to slave. When 1, both privileged and non-privileged transactions are allowed to slave	RW	0x0
22	i2c2	Privilege bit for i2c2. When 0, only privileged transactions are allowed to slave. When 1, both privileged and non-privileged transactions are allowed to slave	RW	0x0
21	i2c1	Privilege bit for i2c1. When 0, only privileged transactions are allowed to slave. When 1, both privileged and non-privileged transactions are allowed to slave	RW	0x0

Bit	Name	Description	Access	Reset
20	i2c0	Privilege bit for i2c0. When 0, only privileged transactions are allowed to slave. When 1, both privileged and non-privileged transactions are allowed to slave	RW	0x0
19	gpio2	Privilege bit for gpio2. When 0, only privileged transactions are allowed to slave. When 1, both privileged and non-privileged transactions are allowed to slave	RW	0x0
18	gpio1	Privilege bit for gpio1. When 0, only privileged transactions are allowed to slave. When 1, both privileged and non-privileged transactions are allowed to slave	RW	0x0
17	gpio0	Privilege bit for gpio0. When 0, only privileged transactions are allowed to slave. When 1, both privileged and non-privileged transactions are allowed to slave	RW	0x0
16	sdmmc	Privilege bit for sdmmc. When 0, only privileged transactions are allowed to slave. When 1, both privileged and non-privileged transactions are allowed to slave	RW	0x0
15	qspi	Privilege bit for qspi. When 0, only privileged transactions are allowed to slave. When 1, both privileged and non-privileged transactions are allowed to slave	RW	0x0
14	emac3	Privilege bit for emac3. When 0, only privileged transactions are allowed to slave. When 1, both privileged and non-privileged transactions are allowed to slave	RW	0x0
13	emac2	Privilege bit for emac2. When 0, only privileged transactions are allowed to slave. When 1, both privileged and non-privileged transactions are allowed to slave	RW	0x0

Bit	Name	Description	Access	Reset
12	emac1	Privilege bit for emac1. When 0, only privileged transactions are allowed to slave. When 1, both privileged and non-privileged transactions are allowed to slave	RW	0x0
11	emac0	Privilege bit for emac0. When 0, only privileged transactions are allowed to slave. When 1, both privileged and non-privileged transactions are allowed to slave	RW	0x0
10	spi_slave1	Privilege bit for spi_slave1. When 0, only privileged transactions are allowed to slave. When 1, both privileged and non-privileged transactions are allowed to slave	RW	0x0
9	spi_slave0	Privilege bit for spi_slave0. When 0, only privileged transactions are allowed to slave. When 1, both privileged and non-privileged transactions are allowed to slave	RW	0x0
8	spi_master1	Privilege bit for spi_master1. When 0, only privileged transactions are allowed to slave. When 1, both privileged and non-privileged transactions are allowed to slave	RW	0x0
7	spi_master0	Privilege bit for spi_master0. When 0, only privileged transactions are allowed to slave. When 1, both privileged and non-privileged transactions are allowed to slave	RW	0x0
6	dma_secure	Privilege bit for dma_secure. When 0, only privileged transactions are allowed to slave. When 1, both privileged and non-privileged transactions are allowed to slave	RW	0x0
5	dma_nonsecure	Privilege bit for dma_nonsecure. When 0, only privileged transactions are allowed to slave. When 1, both privileged and non-privileged transactions are allowed to slave	RW	0x0

Bit	Name	Description	Access	Reset
4	usb1_register	Privilege bit for usb1_register. When 0, only privileged transactions are allowed to slave. When 1, both privileged and non-privileged transactions are allowed to slave	RW	0x0
3	usb0_register	Privilege bit for usb0_register. When 0, only privileged transactions are allowed to slave. When 1, both privileged and non-privileged transactions are allowed to slave	RW	0x0
2	qspi_data	Privilege bit for qspi_data. When 0, only privileged transactions are allowed to slave. When 1, both privileged and non-privileged transactions are allowed to slave	RW	0x0
1	nand_data	Privilege bit for nand_data. When 0, only privileged transactions are allowed to slave. When 1, both privileged and non-privileged transactions are allowed to slave	RW	0x0
0	nand_register	Privilege bit for nand register. When 0, only privileged transactions are allowed to slave. When 1, both privileged and non-privileged transactions are allowed to slave	RW	0x0

l4_priv_set

Sets Region Enable field when written with 1

Module Instance	Base Address	Register Address
noc_l4_priv_l4_priv_filter	0xFFD11000	0xFFD11004

Offset: 0x4

Access: WO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	soc2fpga WO 0x0	lwsoc2fpga WO 0x0	uart1 WO 0x0	uart0 WO 0x0	sp_timer1 WO 0x0	sp_timer0 WO 0x0	i2c4 WO 0x0	i2c3 WO 0x0	i2c2 WO 0x0	i2c1 WO 0x0	i2c0 WO 0x0	gpio2 WO 0x0	gpio1 WO 0x0	gpio0 WO 0x0	sdmmc WO 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
qspi WO 0x0	emac3 0x0	emac2 WO 0x0	emac1 WO 0x0	emac0 WO 0x0	spi_slave1 WO 0x0	spi_slave0 WO 0x0	spi_master1 WO 0x0	spi_master0 WO 0x0	dma_secure WO 0x0	dma_nonsecure WO 0x0	usb1_register WO 0x0	usb0_register WO 0x0	qspi_data WO 0x0	nand_data WO 0x0	nand_register WO 0x0

l4_priv_set Fields

Bit	Name	Description	Access	Reset
30	soc2fpga	Privilege bit for SOC2FPGA. Writing zero has no effect. Writing one will set the privilege bit	WO	0x0
29	lwsoc2fpga	Privilege bit for Lightweight SOC2FPGA. Writing zero has no effect. Writing one will set the privilege bit	WO	0x0
28	uart1	Privilege bit for uart1. Writing zero has no effect. Writing one will set the privilege bit	WO	0x0
27	uart0	Privilege bit for uart0. Writing zero has no effect. Writing one will set the privilege bit	WO	0x0
26	sp_timer1	Privilege bit for sp_timer1. Writing zero has no effect. Writing one will set the privilege bit	WO	0x0
25	sp_timer0	Privilege bit for sp_timer0. Writing zero has no effect. Writing one will set the privilege bit	WO	0x0
24	i2c4	Privilege bit for i2c4. Writing zero has no effect. Writing one will set the privilege bit	WO	0x0
23	i2c3	Privilege bit for i2c3. Writing zero has no effect. Writing one will set the privilege bit	WO	0x0

Bit	Name	Description	Access	Reset
22	i2c2	Privilege bit for i2c2. Writing zero has no effect. Writing one will set the privilege bit	WO	0x0
21	i2c1	Privilege bit for i2c1. Writing zero has no effect. Writing one will set the privilege bit	WO	0x0
20	i2c0	Privilege bit for i2c0. Writing zero has no effect. Writing one will set the privilege bit	WO	0x0
19	gpio2	Privilege bit for gpio2. Writing zero has no effect. Writing one will set the privilege bit	WO	0x0
18	gpio1	Privilege bit for gpio1. Writing zero has no effect. Writing one will set the privilege bit	WO	0x0
17	gpio0	Privilege bit for gpio0. Writing zero has no effect. Writing one will set the privilege bit	WO	0x0
16	sdmmc	Privilege bit for sdmmc. Writing zero has no effect. Writing one will set the privilege bit	WO	0x0
15	qspi	Privilege bit for qspi. Writing zero has no effect. Writing one will set the privilege bit	WO	0x0
14	emac3	Privilege bit for emac3. Writing zero has no effect. Writing one will set the privilege bit	WO	0x0
13	emac2	Privilege bit for emac2. Writing zero has no effect. Writing one will set the privilege bit	WO	0x0
12	emac1	Privilege bit for emac1. Writing zero has no effect. Writing one will set the privilege bit	WO	0x0
11	emac0	Privilege bit for emac0. Writing zero has no effect. Writing one will set the privilege bit	WO	0x0

Bit	Name	Description	Access	Reset
10	spi_slave1	Privilege bit for spi_slave1. Writing zero has no effect. Writing one will set the privilege bit	WO	0x0
9	spi_slave0	Privilege bit for spi_slave0. Writing zero has no effect. Writing one will set the privilege bit	WO	0x0
8	spi_master1	Privilege bit for spi_master1. Writing zero has no effect. Writing one will set the privilege bit	WO	0x0
7	spi_master0	Privilege bit for spi_master0. Writing zero has no effect. Writing one will set the privilege bit	WO	0x0
6	dma_secure	Privilege bit for dma_secure. Writing zero has no effect. Writing one will set the privilege bit	WO	0x0
5	dma_nonsecure	Privilege bit for dma_nonsecure. Writing zero has no effect. Writing one will set the privilege bit	WO	0x0
4	usb1_register	Privilege bit for usb1_register. Writing zero has no effect. Writing one will set the privilege bit	WO	0x0
3	usb0_register	Privilege bit for usb0_register. Writing zero has no effect. Writing one will set the privilege bit	WO	0x0
2	qspi_data	Privilege bit for qspi_data. Writing zero has no effect. Writing one will set the privilege bit	WO	0x0
1	nand_data	Privilege bit for nand_data. Writing zero has no effect. Writing one will set the privilege bit	WO	0x0
0	nand_register	Privilege bit for nand register. Writing zero has no effect. Writing one will set the privilege bit	WO	0x0

l4_priv_clear

Clears Region Enable field when written with 1

Module Instance	Base Address	Register Address
noc_l4_priv_l4_priv_filter	0xFFD11000	0xFFD11008

Offset: 0x8

Access: WO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	soc2fpga WO 0x0	lwsoc2fpga WO 0x0	uart1 WO 0x0	uart0 WO 0x0	sp_timer1 WO 0x0	sp_timer0 WO 0x0	i2c4 WO 0x0	i2c3 WO 0x0	i2c2 WO 0x0	i2c1 WO 0x0	i2c0 WO 0x0	gpio2 WO 0x0	gpio1 WO 0x0	gpio0 WO 0x0	sdmmc WO 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
qspi WO 0x0	emac3 0x0	emac2 WO 0x0	emac1 WO 0x0	emac0 WO 0x0	spi_slave1 WO 0x0	spi_slave0 WO 0x0	spi_master1 WO 0x0	spi_master0 WO 0x0	dma_secure WO 0x0	dma_nonsecure WO 0x0	usb1_register WO 0x0	usb0_register WO 0x0	qspi_data WO 0x0	nand_data WO 0x0	nand_register WO 0x0

l4_priv_clear Fields

Bit	Name	Description	Access	Reset
30	soc2fpga	Privilege bit for SOC2FPGA. Writing zero has no effect. Writing one will clear the privilege bit	WO	0x0
29	lwsoc2fpga	Privilege bit for Lightweight SOC2FPGA. Writing zero has no effect. Writing one will clear the privilege bit	WO	0x0
28	uart1	Privilege bit for uart1. Writing zero has no effect. Writing one will clear the privilege bit	WO	0x0
27	uart0	Privilege bit for uart0. Writing zero has no effect. Writing one will clear the privilege bit	WO	0x0
26	sp_timer1	Privilege bit for sp_timer1. Writing zero has no effect. Writing one will clear the privilege bit	WO	0x0

Bit	Name	Description	Access	Reset
25	sp_timer0	Privilege bit for sp_timer0. Writing zero has no effect. Writing one will clear the privilege bit	WO	0x0
24	i2c4	Privilege bit for i2c4. Writing zero has no effect. Writing one will clear the privilege bit	WO	0x0
23	i2c3	Privilege bit for i2c3. Writing zero has no effect. Writing one will clear the privilege bit	WO	0x0
22	i2c2	Privilege bit for i2c2. Writing zero has no effect. Writing one will clear the privilege bit	WO	0x0
21	i2c1	Privilege bit for i2c1. Writing zero has no effect. Writing one will clear the privilege bit	WO	0x0
20	i2c0	Privilege bit for i2c0. Writing zero has no effect. Writing one will clear the privilege bit	WO	0x0
19	gpio2	Privilege bit for gpio2. Writing zero has no effect. Writing one will clear the privilege bit	WO	0x0
18	gpio1	Privilege bit for gpio1. Writing zero has no effect. Writing one will clear the privilege bit	WO	0x0
17	gpio0	Privilege bit for gpio0. Writing zero has no effect. Writing one will clear the privilege bit	WO	0x0
16	sdmmc	Privilege bit for sdmmc. Writing zero has no effect. Writing one will clear the privilege bit	WO	0x0
15	qspi	Privilege bit for qspi. Writing zero has no effect. Writing one will clear the privilege bit	WO	0x0
14	emac3	Privilege bit for emac3. Writing zero has no effect. Writing one will clear the privilege bit	WO	0x0

Bit	Name	Description	Access	Reset
13	emac2	Privilege bit for emac2. Writing zero has no effect. Writing one will clear the privilege bit	WO	0x0
12	emac1	Privilege bit for emac1. Writing zero has no effect. Writing one will clear the privilege bit	WO	0x0
11	emac0	Privilege bit for emac0. Writing zero has no effect. Writing one will clear the privilege bit	WO	0x0
10	spi_slave1	Privilege bit for spi_slave1. Writing zero has no effect. Writing one will clear the privilege bit	WO	0x0
9	spi_slave0	Privilege bit for spi_slave0. Writing zero has no effect. Writing one will clear the privilege bit	WO	0x0
8	spi_master1	Privilege bit for spi_master1. Writing zero has no effect. Writing one will clear the privilege bit	WO	0x0
7	spi_master0	Privilege bit for spi_master0. Writing zero has no effect. Writing one will clear the privilege bit	WO	0x0
6	dma_secure	Privilege bit for dma_secure. Writing zero has no effect. Writing one will clear the privilege bit	WO	0x0
5	dma_nonsecure	Privilege bit for dma_nonsecure. Writing zero has no effect. Writing one will clear the privilege bit	WO	0x0
4	usb1_register	Privilege bit for usb1_register. Writing zero has no effect. Writing one will clear the privilege bit	WO	0x0
3	usb0_register	Privilege bit for usb0_register. Writing zero has no effect. Writing one will clear the privilege bit	WO	0x0
2	qspi_data	Privilege bit for qspi_data. Writing zero has no effect. Writing one will clear the privilege bit	WO	0x0

Bit	Name	Description	Access	Reset
1	nand_data	Privilege bit for nand_data. Writing zero has no effect. Writing one will clear the privilege bit	WO	0x0
0	nand_register	Privilege bit for nand register. Writing zero has no effect. Writing one will clear the privilege bit	WO	0x0

noc_mpu_m0_MPU_M1toDDRResp_main_RateAdapter Address Map

Module Instance	Base Address	End Address
i_noc_mpu_m0_MPU_M1toDDRResp_main_RateAdapter	0xFFD11100	0xFFD111FF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
MPU_M1toDDRResp_main_RateAdapter_Id_CoreId on page 7-204	0x0	32	RO	0xC8A2B301	
MPU_M1toDDRResp_main_RateAdapter_Id_RevisionId on page 7-205	0x4	32	RO	0x129FF00	
MPU_M1toDDRResp_main_RateAdapter_Rate on page 7-206	0x8	32	RW	0x0	
MPU_M1toDDRResp_main_RateAdapter_Bypass on page 7-207	0xC	32	RW	0x0	

noc_mpu_m0_MPU_M1toDDRResp_main_RateAdapter Summary

Base Address: 0xFFD11100

Register Address Offset	Bit Fields															
i_noc_mpu_m0_MPU_M1toDDRResp_main_RateAdapter																
MPU_M1toDDRResp_main_RateAdapter_Id_CoreId 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	CORECHECKSUM RO 0xC8A2B3															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CORECHECKSUM RO 0xC8A2B3								CORETYPEID RO 0x1							
MPU_M1toDDRResp_main_RateAdapter_Id_RevisionId 0x4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	FLEXNOCID RO 0x129FF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FLEXNOCID RO 0x129FF								USERID RO 0x0							
MPU_M1toDDRResp_main_RateAdapter_Rate 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						RATE RW 0x0									
MPU_M1toDDRResp_main_RateAdapter_Bypass 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															BYPASS RW 0x0

MPU_M1toDDRResp_main_RateAdapter_Id_CoreId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_MPU_M1toDDRResp_main_RateAdapter	0xFFD11100	0xFFD11100

Offset: 0x0

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CORECHECKSUM RO 0xC8A2B3															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CORECHECKSUM RO 0xC8A2B3								CORETYPEID RO 0x1							

MPU_M1toDDRResp_main_RateAdapter_Id_CoreId Fields

Bit	Name	Description	Access	Reset
31:8	CORECHECKSUM	Field containing a checksum of the parameters of the IP.	RO	0xC8A2B3
7:0	CORETYPEID	Field identifying the type of IP.	RO	0x1

MPU_M1toDDRResp_main_RateAdapter_Id_RevisionId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_MPU_M1toDDRResp_main_RateAdapter	0xFFD11100	0xFFD11104

Offset: 0x4

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLEXNOCID RO 0x129FF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLEXNOCID RO 0x129FF								USERID RO 0x0							

MPU_M1toDDRResp_main_RateAdapter_Id_RevisionId Fields

Bit	Name	Description	Access	Reset
31:8	FLEXNOCID	Field containing the build revision of the software used to generate the IP HDL code.	RO	0x129FF
7:0	USERID	Field containing a user defined value, not used anywhere inside the IP itself.	RO	0x0

MPU_M1toDDRResp_main_RateAdapter_Rate

Module Instance	Base Address	Register Address
i_noc_mpu_m0_MPU_M1toDDRResp_main_RateAdapter	0xFFD11100	0xFFD11108

Offset: 0x8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						RATE RW 0x0									

MPU_M1toDDRResp_main_RateAdapter_Rate Fields

Bit	Name	Description	Access	Reset
9:0	RATE	The ratio of outgoing to incoming throughput. This value determines what portion of a received packet will be stored before its head is transmitted. An optimal setting avoids transmitting bubbles, while adding no delay to packets. The ratio is expressed as 256 / (ratio - 1). The value for rate must be rounded up except when ratio is greater than 4, in which case it must be rounded down. For example, a 4:1 ratio of outgoing to incoming throughput would be indicated by value 0x055. Note that throughput is the product of clock frequency x data bus width. A value of 0x000 causes the rate adapter to store a packet until either the entire packet is received or the buffer becomes full.	RW	0x0

MPU_M1toDDRResp_main_RateAdapter_Bypass

Module Instance	Base Address	Register Address
i_noc_mpu_m0_MPU_M1toDDRResp_main_RateAdapter	0xFFD11100	0xFFD1110C

Offset: 0xC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															BYPASS RW 0x0

MPU_M1toDDRResp_main_RateAdapter_Bypass Fields

Bit	Name	Description	Access	Reset
0	BYPASS	Disable the rate adaptation capability. This causes the rate adapter to act as a FIFO by transmitting received words, without delay, as soon as they can be transmitted. This setting is useful when the incoming throughput is equal to or greater than the downstream throughput.	RW	0x0

noc_mpu_m0_MPU_M0_rate_adResp_main_RateAdapter Address Map

Module Instance	Base Address	End Address
i_noc_mpu_m0_MPU_M0_rate_adResp_main_RateAdapter	0xFFD11200	0xFFD112FF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
MPU_M0_rate_adResp_main_RateAdapter_Id_CoreId on page 7-210	0x0	32	RO	0x7AF9A601	
MPU_M0_rate_adResp_main_RateAdapter_Id_RevisionId on page 7-211	0x4	32	RO	0x129FF00	

Register	Offset	Width	Access	Reset Value	Description
MPU_M0_rate_adResp_main_RateAdapter_Rate on page 7-211	0x8	32	RW	0x0	
MPU_M0_rate_adResp_main_RateAdapter_Bypass on page 7-212	0xC	32	RW	0x0	

noc_mpu_m0_MPU_M0_rate_adResp_main_RateAdapter Summary

Base Address: 0xFFD11200

Register Address Offset	Bit Fields															
i_noc_mpu_m0_MPU_M0_rate_adResp_main_RateAdapter																
MPU_M0_rate_adResp_main_RateAdapter_Id_CoreId 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	CORECHECKSUM RO 0x7AF9A6															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CORECHECKSUM RO 0x7AF9A6								CORETYPEID RO 0x1							
MPU_M0_rate_adResp_main_RateAdapter_Id_RevisionId 0x4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	FLEXNOCID RO 0x129FF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FLEXNOCID RO 0x129FF								USERID RO 0x0							
MPU_M0_rate_adResp_main_RateAdapter_Rate 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						RATE RW 0x0									

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MPU_M0_rate_adResp_main_RateAdapter_Bypass	Reserved															
0xC	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															BYPASS RW 0x0

MPU_M0_rate_adResp_main_RateAdapter_Id_CoreId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_MPU_M0_rate_adResp_main_RateAdapter	0xFFD11200	0xFFD11200

Offset: 0x0

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CORECHECKSUM RO 0x7AF9A6															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CORECHECKSUM RO 0x7AF9A6								CORETYPEID RO 0x1							

MPU_M0_rate_adResp_main_RateAdapter_Id_CoreId Fields

Bit	Name	Description	Access	Reset
31:8	CORECHECKSUM	Field containing a checksum of the parameters of the IP.	RO	0x7AF9A6
7:0	CORETYPEID	Field identifying the type of IP.	RO	0x1

MPU_M0_rate_adResp_main_RateAdapter_Id_RevisionId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_MPU_M0_rate_adResp_main_RateAdapter	0xFFD11200	0xFFD11204

Offset: 0x4

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLEXNOCID RO 0x129FF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLEXNOCID RO 0x129FF								USERID RO 0x0							

MPU_M0_rate_adResp_main_RateAdapter_Id_RevisionId Fields

Bit	Name	Description	Access	Reset
31:8	FLEXNOCID	Field containing the build revision of the software used to generate the IP HDL code.	RO	0x129FF
7:0	USERID	Field containing a user defined value, not used anywhere inside the IP itself.	RO	0x0

MPU_M0_rate_adResp_main_RateAdapter_Rate

Module Instance	Base Address	Register Address
i_noc_mpu_m0_MPU_M0_rate_adResp_main_RateAdapter	0xFFD11200	0xFFD11208

Offset: 0x8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						RATE RW 0x0									

MPU_M0_rate_adResp_main_RateAdapter_Rate Fields

Bit	Name	Description	Access	Reset
9:0	RATE	The ratio of outgoing to incoming throughput. This value determines what portion of a received packet will be stored before its head is transmitted. An optimal setting avoids transmitting bubbles, while adding no delay to packets. The ratio is expressed as 256 / (ratio - 1). The value for rate must be rounded up except when ratio is greater than 4, in which case it must be rounded down. For example, a 4:1 ratio of outgoing to incoming throughput would be indicated by value 0x055. Note that throughput is the product of clock frequency x data bus width. A value of 0x000 causes the rate adapter to store a packet until either the entire packet is received or the buffer becomes full.	RW	0x0

MPU_M0_rate_adResp_main_RateAdapter_Bypass

Module Instance	Base Address	Register Address
i_noc_mpu_m0_MPU_M0_rate_adResp_main_RateAdapter	0xFFD11200	0xFFD1120C

Offset: 0xC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															BYPASS RW 0x0

MPU_M0_rate_adResp_main_RateAdapter_Bypass Fields

Bit	Name	Description	Access	Reset
0	BYPASS	Disable the rate adaptation capability. This causes the rate adapter to act as a FIFO by transmitting received words, without delay, as soon as they can be transmitted. This setting is useful when the incoming throughput is equal to or greater than the downstream throughput.	RW	0x0

noc_mpu_m0_L4_MP_rate_ad_main_RateAdapter Address Map

Module Instance	Base Address	End Address
i_noc_mpu_m0_L4_MP_rate_ad_main_RateAdapter	0xFFD11300	0xFFD113FF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
L4_MP_rate_ad_main_RateAdapter_Id_CoreId on page 7-215	0x0	32	RO	0x56875401	

Register	Offset	Width	Access	Reset Value	Description
L4_MP_rate_ad_main_RateAdapter_Id_RevisionId on page 7-216	0x4	32	RO	0x129FF00	
L4_MP_rate_ad_main_RateAdapter_Rate on page 7-216	0x8	32	RW	0x100	
L4_MP_rate_ad_main_RateAdapter_Bypass on page 7-217	0xC	32	RW	0x0	

noc_mpu_m0_L4_MP_rate_ad_main_RateAdapter Summary

Base Address: 0xFFD11300

Register Address Offset	Bit Fields															
i_noc_mpu_m0_L4_MP_rate_ad_main_RateAdapter																
L4_MP_rate_ad_main_RateAdapter_Id_CoreId 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	CORECHECKSUM RO 0x568754															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CORECHECKSUM RO 0x568754								CORETYPEID RO 0x1							
L4_MP_rate_ad_main_RateAdapter_Id_RevisionId 0x4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	FLEXNOCID RO 0x129FF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FLEXNOCID RO 0x129FF								USERID RO 0x0							
L4_MP_rate_ad_main_RateAdapter_Rate 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						RATE RW 0x100									

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
L4_MP_rate_ad_main_RateAdapter_Bypass 0xC	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															BYPASS RW 0x0

L4_MP_rate_ad_main_RateAdapter_Id_CoreId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_L4_MP_rate_ad_main_RateAdapter	0xFFD11300	0xFFD11300

Offset: 0x0

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CORECHECKSUM RO 0x568754															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CORECHECKSUM RO 0x568754								CORETYPEID RO 0x1							

L4_MP_rate_ad_main_RateAdapter_Id_CoreId Fields

Bit	Name	Description	Access	Reset
31:8	CORECHECKSUM	Field containing a checksum of the parameters of the IP.	RO	0x568754
7:0	CORETYPEID	Field identifying the type of IP.	RO	0x1

L4_MP_rate_ad_main_RateAdapter_Id_RevisionId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_L4_MP_rate_ad_main_RateAdapter	0xFFD11300	0xFFD11304

Offset: 0x4

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLEXNOCID RO 0x129FF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLEXNOCID RO 0x129FF								USERID RO 0x0							

L4_MP_rate_ad_main_RateAdapter_Id_RevisionId Fields

Bit	Name	Description	Access	Reset
31:8	FLEXNOCID	Field containing the build revision of the software used to generate the IP HDL code.	RO	0x129FF
7:0	USERID	Field containing a user defined value, not used anywhere inside the IP itself.	RO	0x0

L4_MP_rate_ad_main_RateAdapter_Rate

Module Instance	Base Address	Register Address
i_noc_mpu_m0_L4_MP_rate_ad_main_RateAdapter	0xFFD11300	0xFFD11308

Offset: 0x8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						RATE RW 0x100									

L4_MP_rate_ad_main_RateAdapter_Rate Fields

Bit	Name	Description	Access	Reset
9:0	RATE	The ratio of outgoing to incoming throughput. This value determines what portion of a received packet will be stored before its head is transmitted. An optimal setting avoids transmitting bubbles, while adding no delay to packets. The ratio is expressed as 256 / (ratio - 1). The value for rate must be rounded up except when ratio is greater than 4, in which case it must be rounded down. For example, a 4:1 ratio of outgoing to incoming throughput would be indicated by value 0x055. Note that throughput is the product of clock frequency x data bus width. A value of 0x000 causes the rate adapter to store a packet until either the entire packet is received or the buffer becomes full.	RW	0x100

L4_MP_rate_ad_main_RateAdapter_Bypass

Module Instance	Base Address	Register Address
i_noc_mpu_m0_L4_MP_rate_ad_main_RateAdapter	0xFFD11300	0xFFD1130C

Offset: 0xC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															BYPASS RW 0x0

L4_MP_rate_ad_main_RateAdapter_Bypass Fields

Bit	Name	Description	Access	Reset
0	BYPASS	Disable the rate adaptation capability. This causes the rate adapter to act as a FIFO by transmitting received words, without delay, as soon as they can be transmitted. This setting is useful when the incoming throughput is equal to or greater than the downstream throughput.	RW	0x0

noc_mpu_m0_fpga2soc_rate_ad_main_RateAdapter Address Map

Module Instance	Base Address	End Address
i_noc_mpu_m0_fpga2soc_rate_ad_main_RateAdapter	0xFFD11400	0xFFD114FF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
fpga2soc_rate_ad_main_RateAdapter_Id_CoreId on page 7-220	0x0	32	RO	0x88DEAF01	

Register	Offset	Width	Access	Reset Value	Description
fpga2soc_rate_ad_main_RateAdapter_Id_RevisionId on page 7-221	0x4	32	RO	0x129FF00	
fpga2soc_rate_ad_main_RateAdapter_Rate on page 7-221	0x8	32	RW	0x0	
fpga2soc_rate_ad_main_RateAdapter_Bypass on page 7-222	0xC	32	RW	0x0	

noc_mpu_m0_fpga2soc_rate_ad_main_RateAdapter Summary

Base Address: 0xFFD11400

Register Address Offset	Bit Fields															
i_noc_mpu_m0_fpga2soc_rate_ad_main_RateAdapter																
fpga2soc_rate_ad_main_RateAdapter_Id_CoreId 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	CORECHECKSUM RO 0x88DEAF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CORECHECKSUM RO 0x88DEAF								CORETYPEID RO 0x1							
fpga2soc_rate_ad_main_RateAdapter_Id_RevisionId 0x4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	FLEXNOCID RO 0x129FF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FLEXNOCID RO 0x129FF								USERID RO 0x0							
fpga2soc_rate_ad_main_RateAdapter_Rate 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						RATE RW 0x0									

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
fpga2soc_rate_ad_main_RateAdapter_Bypass	Reserved															
0xC	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															BYPASS RW 0x0

fpga2soc_rate_ad_main_RateAdapter_Id_CoreId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2soc_rate_ad_main_RateAdapter	0xFFD11400	0xFFD11400

Offset: 0x0

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CORECHECKSUM RO 0x88DEAF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CORECHECKSUM RO 0x88DEAF								CORETYPEID RO 0x1							

fpga2soc_rate_ad_main_RateAdapter_Id_CoreId Fields

Bit	Name	Description	Access	Reset
31:8	CORECHECKSUM	Field containing a checksum of the parameters of the IP.	RO	0x88DEAF
7:0	CORETYPEID	Field identifying the type of IP.	RO	0x1

fpga2soc_rate_ad_main_RateAdapter_Id_RevisionId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2soc_rate_ad_main_RateAdapter	0xFFD11400	0xFFD11404

Offset: 0x4

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLEXNOCID RO 0x129FF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLEXNOCID RO 0x129FF								USERID RO 0x0							

fpga2soc_rate_ad_main_RateAdapter_Id_RevisionId Fields

Bit	Name	Description	Access	Reset
31:8	FLEXNOCID	Field containing the build revision of the software used to generate the IP HDL code.	RO	0x129FF
7:0	USERID	Field containing a user defined value, not used anywhere inside the IP itself.	RO	0x0

fpga2soc_rate_ad_main_RateAdapter_Rate

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2soc_rate_ad_main_RateAdapter	0xFFD11400	0xFFD11408

Offset: 0x8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						RATE RW 0x0									

fpga2soc_rate_ad_main_RateAdapter_Rate Fields

Bit	Name	Description	Access	Reset
9:0	RATE	The ratio of outgoing to incoming throughput. This value determines what portion of a received packet will be stored before its head is transmitted. An optimal setting avoids transmitting bubbles, while adding no delay to packets. The ratio is expressed as 256 / (ratio - 1). The value for rate must be rounded up except when ratio is greater than 4, in which case it must be rounded down. For example, a 4:1 ratio of outgoing to incoming throughput would be indicated by value 0x055. Note that throughput is the product of clock frequency x data bus width. A value of 0x000 causes the rate adapter to store a packet until either the entire packet is received or the buffer becomes full.	RW	0x0

fpga2soc_rate_ad_main_RateAdapter_Bypass

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2soc_rate_ad_main_RateAdapter	0xFFD11400	0xFFD1140C

Offset: 0xC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															BYPASS RW 0x0

fpga2soc_rate_ad_main_RateAdapter_Bypass Fields

Bit	Name	Description	Access	Reset
0	BYPASS	Disable the rate adaptation capability. This causes the rate adapter to act as a FIFO by transmitting received words, without delay, as soon as they can be transmitted. This setting is useful when the incoming throughput is equal to or greater than the downstream throughput.	RW	0x0

noc_mpu_m0_L3Tosoc2fpgaResp_main_RateAdapter Address Map

Module Instance	Base Address	End Address
i_noc_mpu_m0_L3Tosoc2fpgaResp_main_RateAdapter	0xFFD11500	0xFFD115FF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
L3Tosoc2fpgaResp_main_RateAdapter_Id_CoreId on page 7-225	0x0	32	RO	0xDB059201	

Register	Offset	Width	Access	Reset Value	Description
L3Tosoc2fpgaResp_main_RateAdapter_Id_RevisionId on page 7-226	0x4	32	RO	0x129FF00	
L3Tosoc2fpgaResp_main_RateAdapter_Rate on page 7-226	0x8	32	RW	0x0	
L3Tosoc2fpgaResp_main_RateAdapter_Bypass on page 7-227	0xC	32	RW	0x0	

noc_mpu_m0_L3Tosoc2fpgaResp_main_RateAdapter Summary

Base Address: 0xFFD11500

Register Address Offset	Bit Fields															
i_noc_mpu_m0_L3Tosoc2fpgaResp_main_RateAdapter																
L3Tosoc2fpgaResp_main_RateAdapter_Id_CoreId 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	CORECHECKSUM RO 0xDB0592															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CORECHECKSUM RO 0xDB0592								CORETYPEID RO 0x1							
L3Tosoc2fpgaResp_main_RateAdapter_Id_RevisionId 0x4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	FLEXNOCID RO 0x129FF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FLEXNOCID RO 0x129FF								USERID RO 0x0							
L3Tosoc2fpgaResp_main_RateAdapter_Rate 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						RATE RW 0x0									

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
L3Tosoc2fpgaResp_main_RateAdapter_Bypass 0xC	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															BYPASS RW 0x0

L3Tosoc2fpgaResp_main_RateAdapter_Id_CoreId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_L3Tosoc2fpgaResp_main_RateAdapter	0xFFD11500	0xFFD11500

Offset: 0x0

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CORECHECKSUM RO 0xDB0592															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CORECHECKSUM RO 0xDB0592								CORETYPEID RO 0x1							

L3Tosoc2fpgaResp_main_RateAdapter_Id_CoreId Fields

Bit	Name	Description	Access	Reset
31:8	CORECHECKSUM	Field containing a checksum of the parameters of the IP.	RO	0xDB0592
7:0	CORETYPEID	Field identifying the type of IP.	RO	0x1

L3Tosoc2fpgaResp_main_RateAdapter_Id_RevisionId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_L3Tosoc2fpgaResp_main_RateAdapter	0xFFD11500	0xFFD11504

Offset: 0x4

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLEXNOCID RO 0x129FF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLEXNOCID RO 0x129FF								USERID RO 0x0							

L3Tosoc2fpgaResp_main_RateAdapter_Id_RevisionId Fields

Bit	Name	Description	Access	Reset
31:8	FLEXNOCID	Field containing the build revision of the software used to generate the IP HDL code.	RO	0x129FF
7:0	USERID	Field containing a user defined value, not used anywhere inside the IP itself.	RO	0x0

L3Tosoc2fpgaResp_main_RateAdapter_Rate

Module Instance	Base Address	Register Address
i_noc_mpu_m0_L3Tosoc2fpgaResp_main_RateAdapter	0xFFD11500	0xFFD11508

Offset: 0x8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						RATE RW 0x0									

L3Tosoc2fpgaResp_main_RateAdapter_Rate Fields

Bit	Name	Description	Access	Reset
9:0	RATE	The ratio of outgoing to incoming throughput. This value determines what portion of a received packet will be stored before its head is transmitted. An optimal setting avoids transmitting bubbles, while adding no delay to packets. The ratio is expressed as 256 / (ratio - 1). The value for rate must be rounded up except when ratio is greater than 4, in which case it must be rounded down. For example, a 4:1 ratio of outgoing to incoming throughput would be indicated by value 0x055. Note that throughput is the product of clock frequency x data bus width. A value of 0x000 causes the rate adapter to store a packet until either the entire packet is received or the buffer becomes full.	RW	0x0

L3Tosoc2fpgaResp_main_RateAdapter_Bypass

Module Instance	Base Address	Register Address
i_noc_mpu_m0_L3Tosoc2fpgaResp_main_RateAdapter	0xFFD11500	0xFFD1150C

Offset: 0xC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															BYPASS RW 0x0

L3Tosoc2fpgaResp_main_RateAdapter_Bypass Fields

Bit	Name	Description	Access	Reset
0	BYPASS	Disable the rate adaptation capability. This causes the rate adapter to act as a FIFO by transmitting received words, without delay, as soon as they can be transmitted. This setting is useful when the incoming throughput is equal to or greater than the downstream throughput.	RW	0x0

noc_mpu_m0_acp_rate_ad_main_RateAdapter Address Map

Module Instance	Base Address	End Address
i_noc_mpu_m0_acp_rate_ad_main_RateAdapter	0xFFD11600	0xFFD11FFF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
acp_rate_ad_main_RateAdapter_Id_CoreId on page 7-230	0x0	32	RO	0x5EDB1801	

Register	Offset	Width	Access	Reset Value	Description
acp_rate_ad_main_RateAdapter_Id_RevisionId on page 7-231	0x4	32	RO	0x129FF00	
acp_rate_ad_main_RateAdapter_Rate on page 7-231	0x8	32	RW	0x0	
acp_rate_ad_main_RateAdapter_Bypass on page 7-232	0xC	32	RW	0x0	

noc_mpu_m0_acp_rate_ad_main_RateAdapter Summary

Base Address: 0xFFD11600

Register Address Offset	Bit Fields															
i_noc_mpu_m0_acp_rate_ad_main_RateAdapter																
acp_rate_ad_main_RateAdapter_Id_CoreId 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	CORECHECKSUM RO 0x5EDB18															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CORECHECKSUM RO 0x5EDB18								CORETYPEID RO 0x1							
acp_rate_ad_main_RateAdapter_Id_RevisionId 0x4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	FLEXNOCID RO 0x129FF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FLEXNOCID RO 0x129FF								USERID RO 0x0							
acp_rate_ad_main_RateAdapter_Rate 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						RATE RW 0x0									

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
acp_rate_ad_main_RateAdapter_Bypass 0xC	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															BYPASS RW 0x0

acp_rate_ad_main_RateAdapter_Id_CoreId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_acp_rate_ad_main_RateAdapter	0xFFD11600	0xFFD11600

Offset: 0x0

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CORECHECKSUM RO 0x5EDB18															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CORECHECKSUM RO 0x5EDB18								CORETYPEID RO 0x1							

acp_rate_ad_main_RateAdapter_Id_CoreId Fields

Bit	Name	Description	Access	Reset
31:8	CORECHECKSUM	Field containing a checksum of the parameters of the IP.	RO	0x5EDB18
7:0	CORETYPEID	Field identifying the type of IP.	RO	0x1

acp_rate_ad_main_RateAdapter_Id_RevisionId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_acp_rate_ad_main_RateAdapter	0xFFD11600	0xFFD11604

Offset: 0x4

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLEXNOCID RO 0x129FF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLEXNOCID RO 0x129FF								USERID RO 0x0							

acp_rate_ad_main_RateAdapter_Id_RevisionId Fields

Bit	Name	Description	Access	Reset
31:8	FLEXNOCID	Field containing the build revision of the software used to generate the IP HDL code.	RO	0x129FF
7:0	USERID	Field containing a user defined value, not used anywhere inside the IP itself.	RO	0x0

acp_rate_ad_main_RateAdapter_Rate

Module Instance	Base Address	Register Address
i_noc_mpu_m0_acp_rate_ad_main_RateAdapter	0xFFD11600	0xFFD11608

Offset: 0x8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						RATE RW 0x0									

acp_rate_ad_main_RateAdapter_Rate Fields

Bit	Name	Description	Access	Reset
9:0	RATE	The ratio of outgoing to incoming throughput. This value determines what portion of a received packet will be stored before its head is transmitted. An optimal setting avoids transmitting bubbles, while adding no delay to packets. The ratio is expressed as 256 / (ratio - 1). The value for rate must be rounded up except when ratio is greater than 4, in which case it must be rounded down. For example, a 4:1 ratio of outgoing to incoming throughput would be indicated by value 0x055. Note that throughput is the product of clock frequency x data bus width. A value of 0x000 causes the rate adapter to store a packet until either the entire packet is received or the buffer becomes full.	RW	0x0

acp_rate_ad_main_RateAdapter_Bypass

Module Instance	Base Address	Register Address
i_noc_mpu_m0_acp_rate_ad_main_RateAdapter	0xFFD11600	0xFFD1160C

Offset: 0xC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															BYPASS RW 0x0

acp_rate_ad_main_RateAdapter_Bypass Fields

Bit	Name	Description	Access	Reset
0	BYPASS	Disable the rate adaptation capability. This causes the rate adapter to act as a FIFO by transmitting received words, without delay, as soon as they can be transmitted. This setting is useful when the incoming throughput is equal to or greater than the downstream throughput.	RW	0x0

noc_mpu_m0_ddr_T_main_Probe Address Map

Module Instance	Base Address	End Address
i_noc_mpu_m0_ddr_T_main_Probe	0xFFD12000	0xFFD123FF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
ddr_T_main_Probe_Id_CoreId on page 7-249	0x0	32	RO	0xFA9ECC06	

Register	Offset	Width	Access	Reset Value	Description
ddr_T_main_Probe_Id_RevisionId on page 7-250	0x4	32	RO	0x129FF00	
ddr_T_main_Probe_MainCtl on page 7-251	0x8	32	RW	0x0	Register MainCtl contains probe global control bits. The register has seven bit fields:
ddr_T_main_Probe_CfgCtl on page 7-253	0xC	32	RW	0x0	
ddr_T_main_Probe_FilterLut on page 7-254	0x14	32	RW	0x0	
ddr_T_main_Probe_TraceAlarmEn on page 7-255	0x18	32	RW	0x0	
ddr_T_main_Probe_TraceAlarmStatus on page 7-256	0x1C	32	RO	0x0	
ddr_T_main_Probe_TraceAlarmClr on page 7-257	0x20	32	RW	0x0	
ddr_T_main_Probe_StatPeriod on page 7-258	0x24	32	RW	0x0	
ddr_T_main_Probe_StatGo on page 7-259	0x28	32	RW	0x0	
ddr_T_main_Probe_StatAlarmMin on page 7-260	0x2C	32	RW	0x0	
ddr_T_main_Probe_StatAlarmMax on page 7-261	0x30	32	RW	0x0	
ddr_T_main_Probe_StatAlarmStatus on page 7-262	0x34	32	RO	0x0	
ddr_T_main_Probe_StatAlarmClr on page 7-263	0x38	32	RW	0x0	
ddr_T_main_Probe_StatAlarmEn on page 7-264	0x3C	32	RW	0x1	

Register	Offset	Width	Access	Reset Value	Description
ddr_T_main_Probe_Filters_0_RouteIdBase on page 7-265	0x44	32	RW	0x0	
ddr_T_main_Probe_Filters_0_RouteIdMask on page 7-265	0x48	32	RW	0x0	
ddr_T_main_Probe_Filters_0_AddrBase_Low on page 7-266	0x4C	32	RW	0x0	
ddr_T_main_Probe_Filters_0_WindowSize on page 7-267	0x54	32	RW	0x0	
ddr_T_main_Probe_Filters_0_SecurityBase on page 7-268	0x58	32	RW	0x0	
ddr_T_main_Probe_Filters_0_SecurityMask on page 7-268	0x5C	32	RW	0x0	
ddr_T_main_Probe_Filters_0_Opcode on page 7-269	0x60	32	RW	0x0	Packet Probe register Opcode is a 4-bit register that selects candidate packets based on packet opcodes (0 disables the filter):
ddr_T_main_Probe_Filters_0_Status on page 7-270	0x64	32	RW	0x0	Register Status is 2-bit register that selects candidate packets based on packet status.
ddr_T_main_Probe_Filters_0_Length on page 7-271	0x68	32	RW	0x0	
ddr_T_main_Probe_Filters_0_Urgency on page 7-272	0x6C	32	RW	0x0	
ddr_T_main_Probe_Filters_1_RouteIdBase on page 7-273	0x80	32	RW	0x0	

Register	Offset	Width	Access	Reset Value	Description
ddr_T_main_Probe_Filters_1_RouteIdMask on page 7-273	0x84	32	RW	0x0	
ddr_T_main_Probe_Filters_1_AddrBase_Low on page 7-274	0x88	32	RW	0x0	
ddr_T_main_Probe_Filters_1_WindowSize on page 7-275	0x90	32	RW	0x0	
ddr_T_main_Probe_Filters_1_SecurityBase on page 7-276	0x94	32	RW	0x0	
ddr_T_main_Probe_Filters_1_SecurityMask on page 7-276	0x98	32	RW	0x0	
ddr_T_main_Probe_Filters_1_Opcode on page 7-277	0x9C	32	RW	0x0	Packet Probe register Opcode is a 4-bit register that selects candidate packets based on packet opcodes (0 disables the filter):
ddr_T_main_Probe_Filters_1_Status on page 7-278	0xA0	32	RW	0x0	Register Status is 2-bit register that selects candidate packets based on packet status.
ddr_T_main_Probe_Filters_1_Length on page 7-279	0xA4	32	RW	0x0	
ddr_T_main_Probe_Filters_1_Urgency on page 7-280	0xA8	32	RW	0x0	
ddr_T_main_Probe_Filters_2_RouteIdBase on page 7-281	0xBC	32	RW	0x0	
ddr_T_main_Probe_Filters_2_RouteIdMask on page 7-281	0xC0	32	RW	0x0	

Register	Offset	Width	Access	Reset Value	Description
ddr_T_main_Probe_Filters_2_AddrBase_Low on page 7-282	0xC4	32	RW	0x0	
ddr_T_main_Probe_Filters_2_WindowSize on page 7-283	0xCC	32	RW	0x0	
ddr_T_main_Probe_Filters_2_SecurityBase on page 7-284	0xD0	32	RW	0x0	
ddr_T_main_Probe_Filters_2_SecurityMask on page 7-284	0xD4	32	RW	0x0	
ddr_T_main_Probe_Filters_2_Opcode on page 7-285	0xD8	32	RW	0x0	Packet Probe register Opcode is a 4-bit register that selects candidate packets based on packet opcodes (0 disables the filter):
ddr_T_main_Probe_Filters_2_Status on page 7-286	0xDC	32	RW	0x0	Register Status is 2-bit register that selects candidate packets based on packet status.
ddr_T_main_Probe_Filters_2_Length on page 7-287	0xE0	32	RW	0x0	
ddr_T_main_Probe_Filters_2_Urgency on page 7-288	0xE4	32	RW	0x0	
ddr_T_main_Probe_Filters_3_RouteIdBase on page 7-289	0xF8	32	RW	0x0	
ddr_T_main_Probe_Filters_3_RouteIdMask on page 7-289	0xFC	32	RW	0x0	
ddr_T_main_Probe_Filters_3_AddrBase_Low on page 7-290	0x100	32	RW	0x0	

Register	Offset	Width	Access	Reset Value	Description
ddr_T_main_Probe_Filters_3_WindowSize on page 7-291	0x108	32	RW	0x0	
ddr_T_main_Probe_Filters_3_SecurityBase on page 7-292	0x10C	32	RW	0x0	
ddr_T_main_Probe_Filters_3_SecurityMask on page 7-292	0x110	32	RW	0x0	
ddr_T_main_Probe_Filters_3_Opcode on page 7-293	0x114	32	RW	0x0	Packet Probe register Opcode is a 4-bit register that selects candidate packets based on packet opcodes (0 disables the filter):
ddr_T_main_Probe_Filters_3_Status on page 7-294	0x118	32	RW	0x0	Register Status is 2-bit register that selects candidate packets based on packet status.
ddr_T_main_Probe_Filters_3_Length on page 7-295	0x11C	32	RW	0x0	
ddr_T_main_Probe_Filters_3_Urgency on page 7-296	0x120	32	RW	0x0	
ddr_T_main_Probe_Counters_0_Src on page 7-297	0x138	32	RW	0x0	Register CntSrc indicates the event source used to increment the counter. Unassigned values (non-existing Press level or ExtEvent index, or unimplemented Filter) are equivalent to OFF.
ddr_T_main_Probe_Counters_0_AlarmMode on page 7-297	0x13C	32	RW	0x0	

Register	Offset	Width	Access	Reset Value	Description
ddr_T_main_Probe_Counters_0_Val on page 7-298	0x140	32	RO	0x0	
ddr_T_main_Probe_Counters_1_Src on page 7-299	0x14C	32	RW	0x0	Register CntSrc indicates the event source used to increment the counter. Unassigned values (non-existing Press level or ExtEvent index, or unimplemented Filter) are equivalent to OFF.
ddr_T_main_Probe_Counters_1_AlarmMode on page 7-300	0x150	32	RW	0x0	
ddr_T_main_Probe_Counters_1_Val on page 7-301	0x154	32	RO	0x0	

noc_mpu_m0_ddr_T_main_Probe Summary

Base Address: 0xFFD12000

Register Address Offset	Bit Fields															
i_noc_mpu_m0_ddr_T_main_Probe																
ddr_T_main_Probe_Id_CoreId 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	CORECHECKSUM RO 0xFA9ECC															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CORECHECKSUM RO 0xFA9ECC								CORETYPEID RO 0x6							

Register Address Offset	Bit Fields															
ddr_T_main_Probe_Id_RevisionId 0x4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	FLEXNOCID RO 0x129FF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FLEXNOCID RO 0x129FF								USERID RO 0x0							
ddr_T_main_Probe_MainCtl 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								FILTBYTEALWAYSCHAINABLEEN RW 0x0	INTRUSIVEMODE RW 0x0	STATCONDDUMP RW 0x0	ALARMEN RW 0x0	STATEN RW 0x0	PAYLOADEN RW 0x0	TRACEEN RW 0x0	ERREN RW 0x0
ddr_T_main_Probe_CfgCtl 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														ACTIVE RO 0x0	GLOBALEN RW 0x0
ddr_T_main_Probe_FilterLut 0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FILTERLUT RW 0x0															
ddr_T_main_Probe_TraceAlarmEn 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved											TRACEALARMEN RW 0x0				
ddr_T_main_Probe_TraceAlarmStatus 0x1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved											TRACEALARMSTATUS RO 0x0				

Register Address Offset	Bit Fields															
ddr_T_main_Probe_TraceAlarmClr 0x20	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved											TRACEALARMCLR RW 0x0				
ddr_T_main_Probe_StatPeriod 0x24	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved											STATPERIOD RW 0x0				
ddr_T_main_Probe_StatGo 0x28	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															STATGO RW 0x0
ddr_T_main_Probe_StatAlarmMin 0x2C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	STATALARMMIN RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	STATALARMMIN RW 0x0															
ddr_T_main_Probe_StatAlarmMax 0x30	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	STATALARMMAX RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	STATALARMMAX RW 0x0															
ddr_T_main_Probe_StatAlarmStatus 0x34	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															STATALARMSTATUS RO 0x0

Register Address Offset	Bit Fields															
ddr_T_main_Probe_StatAlarmClr 0x38	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															STATALARMCLR RW 0x0
ddr_T_main_Probe_StatAlarmEn 0x3C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															STATALARMEN RW 0x1
ddr_T_main_Probe_Filters_0_RouteIdBase 0x44	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved													FILTERS_0_ROUTEID-BASE RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FILTERS_0_ROUTEIDBASE RW 0x0															
ddr_T_main_Probe_Filters_0_RouteIdMask 0x48	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved													FILTERS_0_ROUTEID-MASK RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FILTERS_0_ROUTEIDMASK RW 0x0															
ddr_T_main_Probe_Filters_0_AddrBase_Low 0x4C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	FILTERS_0_ADDRBASE_LOW RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FILTERS_0_ADDRBASE_LOW RW 0x0															
ddr_T_main_Probe_Filters_0_WindowSize 0x54	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved										FILTERS_0_WINDOWSIZE RW 0x0					

Register Address Offset	Bit Fields															
ddr_T_main_Probe_Filters_0_SecurityBase 0x58	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													FILTERS_0_SECURITY-BASE RW 0x0		
ddr_T_main_Probe_Filters_0_SecurityMask 0x5C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													FILTERS_0_SECURITY-MASK RW 0x0		
ddr_T_main_Probe_Filters_0_Opcode 0x60	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved												URGEN RW 0x0	LOCKEN RW 0x0	WREN RW 0x0	RDEN RW 0x0
ddr_T_main_Probe_Filters_0_Status 0x64	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														RSPEN RW 0x0	REQEN RW 0x0
ddr_T_main_Probe_Filters_0_Length 0x68	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved												FILTERS_0_LENGTH RW 0x0			
ddr_T_main_Probe_Filters_0_Urgency 0x6C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														FILTERS_0_URGENCY RW 0x0	

Register Address Offset	Bit Fields															
ddr_T_main_Probe_Filters_1_RouteIdBase 0x80	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved													FILTERS_1_ROUTEID-BASE RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FILTERS_1_ROUTEIDBASE RW 0x0															
ddr_T_main_Probe_Filters_1_RouteIdMask 0x84	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved													FILTERS_1_ROUTEID-MASK RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FILTERS_1_ROUTEIDMASK RW 0x0															
ddr_T_main_Probe_Filters_1_AddrBase_Low 0x88	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	FILTERS_1_ADDRBASE_LOW RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FILTERS_1_ADDRBASE_LOW RW 0x0															
ddr_T_main_Probe_Filters_1_WindowSize 0x90	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved										FILTERS_1_WINDOWSIZE RW 0x0					
ddr_T_main_Probe_Filters_1_SecurityBase 0x94	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													FILTERS_1_SECURITY-BASE RW 0x0		
ddr_T_main_Probe_Filters_1_SecurityMask 0x98	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													FILTERS_1_SECURITY-MASK RW 0x0		

Register Address Offset	Bit Fields															
ddr_T_main_Probe_Filters_1_Opcode 0x9C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved												URGEN RW 0x0	LOCKEN RW 0x0	WREN RW 0x0	RDEN RW 0x0
ddr_T_main_Probe_Filters_1_Status 0xA0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														RSPEN RW 0x0	REQEN RW 0x0
ddr_T_main_Probe_Filters_1_Length 0xA4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved												FILTERS_1_LENGTH RW 0x0			
ddr_T_main_Probe_Filters_1_Urgency 0xA8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														FILTERS_1_URGENCY RW 0x0	
ddr_T_main_Probe_Filters_2_RouteIdBase 0xBC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved													FILTERS_2_ROUTEID-BASE RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FILTERS_2_ROUTEIDBASE RW 0x0															
ddr_T_main_Probe_Filters_2_RouteIdMask 0xC0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved													FILTERS_2_ROUTEID-MASK RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FILTERS_2_ROUTEIDMASK RW 0x0															

Register Address Offset	Bit Fields															
ddr_T_main_Probe_Filters_2_AddrBase_Low 0xC4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	FILTERS_2_ADDRBASE_LOW RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FILTERS_2_ADDRBASE_LOW RW 0x0															
ddr_T_main_Probe_Filters_2_WindowSize 0xCC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved										FILTERS_2_WINDOWSIZE RW 0x0					
ddr_T_main_Probe_Filters_2_SecurityBase 0xD0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													FILTERS_2_SECURITY-BASE RW 0x0		
ddr_T_main_Probe_Filters_2_SecurityMask 0xD4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													FILTERS_2_SECURITY-MASK RW 0x0		
ddr_T_main_Probe_Filters_2_Opcode 0xD8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved												URGEN RW 0x0	LOCKEN RW 0x0	WREN RW 0x0	RDEN RW 0x0
ddr_T_main_Probe_Filters_2_Status 0xDC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														RSPEN RW 0x0	REQEN RW 0x0

Register Address Offset	Bit Fields															
ddr_T_main_Probe_Filters_2_Length 0xE0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved												FILTERS_2_LENGTH RW 0x0			
ddr_T_main_Probe_Filters_2_Urgency 0xE4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														FILTERS_2_URGENCY RW 0x0	
ddr_T_main_Probe_Filters_3_RouteIdBase 0xF8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved													FILTERS_3_ROUTEID-BASE RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FILTERS_3_ROUTEIDBASE RW 0x0															
ddr_T_main_Probe_Filters_3_RouteIdMask 0xFC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved													FILTERS_3_ROUTEID-MASK RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FILTERS_3_ROUTEIDMASK RW 0x0															
ddr_T_main_Probe_Filters_3_AddrBase_Low 0x100	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	FILTERS_3_ADDRBASE_LOW RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FILTERS_3_ADDRBASE_LOW RW 0x0															
ddr_T_main_Probe_Filters_3_WindowSize 0x108	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved										FILTERS_3_WINDOWSIZE RW 0x0					

Register Address Offset	Bit Fields															
ddr_T_main_Probe_Filters_3_SecurityBase 0x10C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													FILTERS_3_SECURITY-BASE RW 0x0		
ddr_T_main_Probe_Filters_3_SecurityMask 0x110	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													FILTERS_3_SECURITY-MASK RW 0x0		
ddr_T_main_Probe_Filters_3_Opcode 0x114	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved												URGEN RW 0x0	LOCKEN RW 0x0	WREN RW 0x0	RDEN RW 0x0
ddr_T_main_Probe_Filters_3_Status 0x118	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														RSPEN RW 0x0	REQEN RW 0x0
ddr_T_main_Probe_Filters_3_Length 0x11C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved												FILTERS_3_LENGTH RW 0x0			
ddr_T_main_Probe_Filters_3_Urgency 0x120	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														FILTERS_3_URGENCY RW 0x0	

Register Address Offset	Bit Fields															
ddr_T_main_Probe_Counters_0_Src 0x138	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved											INTEVENT RW 0x0				
ddr_T_main_Probe_Counters_0_AlarmMode 0x13C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														COUNTERS_0_ALARMMODE RW 0x0	
ddr_T_main_Probe_Counters_0_Val 0x140	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	COUNTERS_0_VAL RO 0x0															
ddr_T_main_Probe_Counters_1_Src 0x14C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved											INTEVENT RW 0x0				
ddr_T_main_Probe_Counters_1_AlarmMode 0x150	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														COUNTERS_1_ALARMMODE RW 0x0	
ddr_T_main_Probe_Counters_1_Val 0x154	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	COUNTERS_1_VAL RO 0x0															

ddr_T_main_Probe_Id_CoreId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ddr_T_main_Probe	0xFFD12000	0xFFD12000

Offset: 0x0

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CORECHECKSUM RO 0xFA9ECC															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CORECHECKSUM RO 0xFA9ECC								CORETYPEID RO 0x6							

ddr_T_main_Probe_Id_CoreId Fields

Bit	Name	Description	Access	Reset
31:8	CORECHECKSUM	Field containing a checksum of the parameters of the IP.	RO	0xFA9ECC
7:0	CORETYPEID	Field identifying the type of IP.	RO	0x6

ddr_T_main_Probe_Id_RevisionId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ddr_T_main_Probe	0xFFD12000	0xFFD12004

Offset: 0x4

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLEXNOCID RO 0x129FF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLEXNOCID RO 0x129FF								USERID RO 0x0							

ddr_T_main_Probe_Id_RevisionId Fields

Bit	Name	Description	Access	Reset
31:8	FLEXNOCID	Field containing the build revision of the software used to generate the IP HDL code.	RO	0x129FF
7:0	USERID	Field containing a user defined value, not used anywhere inside the IP itself.	RO	0x0

ddr_T_main_Probe_MainCtl

Register MainCtl contains probe global control bits. The register has seven bit fields:

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ddr_T_main_Probe	0xFFD12000	0xFFD12008

Offset: 0x8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								FILTBYTEALWAYSCHAINABLEEN RW 0x0	INTRUSIVEMODE RW 0x0	STATCONDDUMP RW 0x0	ALARMEN RW 0x0	STATEN RW 0x0	PAYLOADEN RW 0x0	TRACEEN RW 0x0	ERREN RW 0x0

ddr_T_main_Probe_MainCtl Fields

Bit	Name	Description	Access	Reset
7	FILTBYTEALWAYSCHAINABLEEN	When set to 0, filters are mapped to all statistic counters when counting bytes or enabled bytes. Therefore, only filter events mapped to even counters can be counted using a pair of chained counters.When set to 1, filters are mapped only to even statistic counters when counting bytes or enabled bytes. Thus events from any filter can be counted using a pair of chained counters.	RW	0x0
6	INTRUSIVEMODE	When set to 1, register field IntrusiveMode enables trace operation in Intrusive flow-control mode. When set to 0, the register enables trace operation in Overflow flow-control mode	RW	0x0
5	STATCONDDUMP	When set, register field StatCond-Dump enables the dump of a statistics frame to the range of counter values set for registers StatAlarmMin, StatAlarmMax, and AlarmMode. This field also renders register StatAlarmStatus inopera-tive. When parameter statisticsCoun-terAlarm is set to False, the StatCondDump register bit is reserved.	RW	0x0

Bit	Name	Description	Access	Reset
4	ALARMEN	When set, register field AlarmEn enables the probe to collect alarm-related information. When the register field bit is null, both TraceAlarm and StatAlarm outputs are driven to 0.	RW	0x0
3	STATEN	When set to 1, register field StatEn enables statistics profiling. The probe sendS statistics results to the output for signal ObsTx. All statistics counters are cleared when the StatEn bit goes from 0 to 1. When set to 0, counters are disabled.	RW	0x0
2	PAYLOADEN	Register field PayloadEn, when set to 1, enables traces to contain headers and payload. When set ot 0, only headers are reported.	RW	0x0
1	TRACEEN	Register field TraceEn enables the probe to send filtered packets (Trace) on the ObsTx observation output.	RW	0x0
0	ERREN	Register field ErrEn enables the probe to send on the ObsTx output any packet with Error status, independently of filtering mechanisms, thus constituting a simple supplementary global filter.	RW	0x0

ddr_T_main_Probe_CfgCtl

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ddr_T_main_Probe	0xFFD12000	0xFFD1200C

Offset: 0xC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														ACTIVE RO 0x0	GLOBALEN RW 0x0

ddr_T_main_Probe_CfgCtl Fields

Bit	Name	Description	Access	Reset
1	ACTIVE		RO	0x0
0	GLOBALEN		RW	0x0

ddr_T_main_Probe_FilterLut

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ddr_T_main_Probe	0xFFD12000	0xFFD12014

Offset: 0x14

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FILTERLUT RW 0x0															

ddr_T_main_Probe_FilterLut Fields

Bit	Name	Description	Access	Reset
15:0	FILTERLUT	Register FilterLut contains a look-up table that is used to combine filter outputs in order to trace packets. Packet tracing is enabled when the FilterLut bit of index (FNout ... F0out) is equal to 1.The number of bits in register FilterLut is determined by the setting for parameter nFilter, calculated as 2**nFilter.When parameter nFilter is set to None, FilterLut is reserved.	RW	0x0

ddr_T_main_Probe_TraceAlarmEn

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ddr_T_main_Probe	0xFFD12000	0xFFD12018

Offset: 0x18

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved											TRACEALARMEN RW 0x0				

ddr_T_main_Probe_TraceAlarmEn Fields

Bit	Name	Description	Access	Reset
4:0	TRACEALARMEN	Register TraceAlarmEn controls which lookup table or filter can set the TraceAlarm signal output once the trace alarm status is set. The number of bits in register TraceAlarmEn is determined by the value set for parameter nFilter + 1.Bit nFilter controls the lookup table output, and bits nFilter:0 control the corresponding filter output. When parameter nFilter is set to None, TraceAlarmEn is reserved.	RW	0x0

ddr_T_main_Probe_TraceAlarmStatus

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ddr_T_main_Probe	0xFFD12000	0xFFD1201C

Offset: 0x1C

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved											TRACEALARMSTATUS RO 0x0				

ddr_T_main_Probe_TraceAlarmStatus Fields

Bit	Name	Description	Access	Reset
4:0	TRACEALARMSTATUS	Register TraceAlarmStatus is a read-only register that indicates which lookup table or filter has been matched by a packet, independently of register TraceAlarmEn bit configuration. The number of bits in TraceAlarmStatus is determined by the value set for parameter nFilter + 1.When nFilter is set to None, TraceAlarmStatus is reserved.	RO	0x0

ddr_T_main_Probe_TraceAlarmClr

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ddr_T_main_Probe	0xFFD12000	0xFFD12020

Offset: 0x20

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved											TRACEALARMCLR RW 0x0				

ddr_T_main_Probe_TraceAlarmClr Fields

Bit	Name	Description	Access	Reset
4:0	TRACEALARMCLR	Setting a bit to 1 in register TraceAlarmClr clears the corresponding bit in register TraceAlarmStatus.The number of bits in register TraceAlarmClr is equal to (nFilter + 1). When nFilter is set to 0, TraceAlarmClr is reserved.NOTE The written value is not stored in TraceAlarmClr. A read always returns 0.	RW	0x0

ddr_T_main_Probe_StatPeriod

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ddr_T_main_Probe	0xFFD12000	0xFFD12024

Offset: 0x24

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved											STATPERIOD RW 0x0				

ddr_T_main_Probe_StatPeriod Fields

Bit	Name	Description	Access	Reset
4:0	STATPERIOD	Register StatPeriod is a 5-bit register that sets a period, within a range of 2 cycles to 2 gigacycles, during which statistics are collected before being dumped automatically. Setting the register implicitly enables automatic mode operation for statistics collection. The period is calculated with the formula: N_Cycle = 2**StatPeriodWhen register StatPeriod is set to its default value 0, automatic dump mode is disabled, and register StatGo is activated for manual mode operation. Note: When parameter statisticsCollection is set to False, StatPeriod is reserved.	RW	0x0

ddr_T_main_Probe_StatGo

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ddr_T_main_Probe	0xFFD12000	0xFFD12028

Offset: 0x28

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															STATGO RW 0x0

ddr_T_main_Probe_StatGo Fields

Bit	Name	Description	Access	Reset
0	STATGO	Writing a 1 to the 1-bit pulse register StatGo generates a statistics dump.The register is active when statistics collection operates in manual mode, that is, when register StatPeriod is set to 0.NOTE The written value is not stored in StatGo. A read always returns 0.	RW	0x0

ddr_T_main_Probe_StatAlarmMin

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ddr_T_main_Probe	0xFFD12000	0xFFD1202C

Offset: 0x2C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
STATALARMMIN RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STATALARMMIN RW 0x0															

ddr_T_main_Probe_StatAlarmMin Fields

Bit	Name	Description	Access	Reset
31:0	STATALARMMIN	Register StatAlarmMin contains the minimum count value used in statistics alarm comparisons. The number of bits is equal to twice the value set forparameter wStatisticsCounter. When parameter statisticsCounterAlarm is set to False, StatAlarmMin is reserved.	RW	0x0

ddr_T_main_Probe_StatAlarmMax

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ddr_T_main_Probe	0xFFD12000	0xFFD12030

Offset: 0x30

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
STATALARMMAX RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STATALARMMAX RW 0x0															

ddr_T_main_Probe_StatAlarmMax Fields

Bit	Name	Description	Access	Reset
31:0	STATALARMMAX	Register StatAlarmMax contains the maximum count value used in statistics alarm comparisons.The number of bits is equal to twice the value set for parameter wStatisticsCounter. When parameter statisticsCounterAlarm is set to False, StatAlarmMax is reserved.	RW	0x0

ddr_T_main_Probe_StatAlarmStatus

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ddr_T_main_Probe	0xFFD12000	0xFFD12034

Offset: 0x34

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															STATALARMSTATUS RO 0x0

ddr_T_main_Probe_StatAlarmStatus Fields

Bit	Name	Description	Access	Reset
0	STATALARMSTATUS	Register StatAlarmStatus is a read-only 1-bit register indicating that at least one statistics counter has exceeded the programmed values for registers StatAlarmMin or StatAlarmMax. Output signal StatAlarm is equal to the values stored in register MainCtl fields StatAlarmStatus and AlarmEn. When parameter statisticsCounterAlarm is set to False, StatAlarmStatus is reserved.	RO	0x0

ddr_T_main_Probe_StatAlarmClr

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ddr_T_main_Probe	0xFFD12000	0xFFD12038

Offset: 0x38

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															STATALARMCLR RW 0x0

ddr_T_main_Probe_StatAlarmClr Fields

Bit	Name	Description	Access	Reset
0	STATALARMCLR	Register StatAlarmClr is a 1-bit register. Writing a 1 to this register clears the StatAlarmStatus register bit.When parameter statisticsCounterAlarm is set to False, StatAlarmClr is reserved.NOTE The written value is not stored in StatAlarmClr. A read always returns 0.	RW	0x0

ddr_T_main_Probe_StatAlarmEn

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ddr_T_main_Probe	0xFFD12000	0xFFD1203C

Offset: 0x3C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															STATALARMEN RW 0x1

ddr_T_main_Probe_StatAlarmEn Fields

Bit	Name	Description	Access	Reset
0	STATALARMEN	Register StatAlarmEn is a 1-bit register. When set to 0 it masks StatAlarm and CtiTrigOut(1) signal interrupts.	RW	0x1

ddr_T_main_Probe_Filters_0_RouteIdBase

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ddr_T_main_Probe	0xFFD12000	0xFFD12044

Offset: 0x44

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													FILTERS_0_ROUTEIDBASE RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FILTERS_0_ROUTEIDBASE RW 0x0															

ddr_T_main_Probe_Filters_0_RouteIdBase Fields

Bit	Name	Description	Access	Reset
18:0	FILTERS_0_ROUTEIDBASE	Register RouteIdBase contains the RouteId-lsbFilterRouteId bits base used to filter packets.	RW	0x0

ddr_T_main_Probe_Filters_0_RouteIdMask

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ddr_T_main_Probe	0xFFD12000	0xFFD12048

Offset: 0x48

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													FILTERS_0_ROUTEIDMASK RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FILTERS_0_ROUTEIDMASK RW 0x0															

ddr_T_main_Probe_Filters_0_RouteIdMask Fields

Bit	Name	Description	Access	Reset
18:0	FILTERS_0_ROUTEIDMASK	Register RouteIdMask contains the RouteId-lsbFilterRouteId mask used to filter packets. A packet is a candidate when packet.RouteId>> lsbFilterRouteId & RouteIdMask = RouteIdBase & RouteIdMask.	RW	0x0

ddr_T_main_Probe_Filters_0_AddrBase_Low

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ddr_T_main_Probe	0xFFD12000	0xFFD1204C

Offset: 0x4C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FILTERS_0_ADDRBASE_LOW RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FILTERS_0_ADDRBASE_LOW RW 0x0															

ddr_T_main_Probe_Filters_0_AddrBase_Low Fields

Bit	Name	Description	Access	Reset
31:0	FILTERS_0_ADDRBASE_LOW	Address LSB register.	RW	0x0

ddr_T_main_Probe_Filters_0_WindowSize

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ddr_T_main_Probe	0xFFD12000	0xFFD12054

Offset: 0x54

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										FILTERS_0_WINDOWSIZE RW 0x0					

ddr_T_main_Probe_Filters_0_WindowSize Fields

Bit	Name	Description	Access	Reset
5:0	FILTERS_0_WINDOWSIZE	Register WindowSize contains the encoded address mask used to filter packets. The effective Mask value is equal to ~(2max(WindowSize, packet.Len) - 1). A packet is a candidate when packet.Addr & Mask = AddrBase & Mask. This allows filteringof packets having an intersection with the AddrBase/ WindowSize burst aligned region, even if the region is smaller than the packet.	RW	0x0

ddr_T_main_Probe_Filters_0_SecurityBase

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ddr_T_main_Probe	0xFFD12000	0xFFD12058

Offset: 0x58

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													FILTERS_0_SECURITY-BASE RW 0x0		

ddr_T_main_Probe_Filters_0_SecurityBase Fields

Bit	Name	Description	Access	Reset
2:0	FILTERS_0_ SECURITYBASE	Register SecurityBase contains the security base used to filter packets.	RW	0x0

ddr_T_main_Probe_Filters_0_SecurityMask

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ddr_T_main_Probe	0xFFD12000	0xFFD1205C

Offset: 0x5C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													FILTERS_0_SECURITY-MASK RW 0x0		

ddr_T_main_Probe_Filters_0_SecurityMask Fields

Bit	Name	Description	Access	Reset
2:0	FILTERS_0_SECURITYMASK	Register SecurityMask is contains the security mask used to filter packets. A packet is a candidate when: packet.Security & Security-Mask = SecurityBase & SecurityMasks.	RW	0x0

ddr_T_main_Probe_Filters_0_Opcode

Packet Probe register Opcode is a 4-bit register that selects candidate packets based on packet opcodes (0 disables the filter):

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ddr_T_main_Probe	0xFFD12000	0xFFD12060

Offset: 0x60

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												URGEN RW 0x0	LOCKEN RW 0x0	WREN RW 0x0	RDEN RW 0x0

ddr_T_main_Probe_Filters_0_Opcode Fields

Bit	Name	Description	Access	Reset
3	URGEN	Selects URG packets (urgency).	RW	0x0
2	LOCKEN	Selects RDX-WR, RDL, WRC and Linked sequence.	RW	0x0
1	WREN	Selects WR packets.	RW	0x0
0	RDEN	Selects RD packets.	RW	0x0

ddr_T_main_Probe_Filters_0_Status

Register Status is 2-bit register that selects candidate packets based on packet status.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ddr_T_main_Probe	0xFFD12000	0xFFD12064

Offset: 0x64

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														RSPEN RW 0x0	REQEN RW 0x0

ddr_T_main_Probe_Filters_0_Status Fields

Bit	Name	Description	Access	Reset
1	RSPEN	Selects RSP and FAIL-CONT status packets.	RW	0x0
0	REQEN	Selects REQ status packets.	RW	0x0

ddr_T_main_Probe_Filters_0_Length

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ddr_T_main_Probe	0xFFD12000	0xFFD12068

Offset: 0x68

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												FILTERS_0_LENGTH RW 0x0			

ddr_T_main_Probe_Filters_0_Length Fields

Bit	Name	Description	Access	Reset
3:0	FILTERS_0_LENGTH	Register Length is 4-bit register that selects candidate packets if their number of bytes is less than or equal to 2**Length.	RW	0x0

ddr_T_main_Probe_Filters_0_Urgency

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ddr_T_main_Probe	0xFFD12000	0xFFD1206C

Offset: 0x6C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														FILTERS_0_URGENCY RW 0x0	

ddr_T_main_Probe_Filters_0_Urgency Fields

Bit	Name	Description	Access	Reset
1:0	FILTERS_0_URGENCY	Register Urgency contains the minimum urgency level used to filter packets. A packet is a candidate when its socket urgency is greater than or equal to the urgency specified in the register.	RW	0x0

ddr_T_main_Probe_Filters_1_RouteIdBase

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ddr_T_main_Probe	0xFFD12000	0xFFD12080

Offset: 0x80

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													FILTERS_1_ROUTEIDBASE RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FILTERS_1_ROUTEIDBASE RW 0x0															

ddr_T_main_Probe_Filters_1_RouteIdBase Fields

Bit	Name	Description	Access	Reset
18:0	FILTERS_1_ROUTEIDBASE	Register RouteIdBase contains the RouteId-lsbFilterRouteId bits base used to filter packets.	RW	0x0

ddr_T_main_Probe_Filters_1_RouteIdMask

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ddr_T_main_Probe	0xFFD12000	0xFFD12084

Offset: 0x84

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													FILTERS_1_ROUTEIDMASK RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FILTERS_1_ROUTEIDMASK RW 0x0															

ddr_T_main_Probe_Filters_1_RouteIdMask Fields

Bit	Name	Description	Access	Reset
18:0	FILTERS_1_ROUTEIDMASK	Register RouteIdMask contains the RouteId-lsbFilterRouteId mask used to filter packets. A packet is a candidate when packet.RouteId>> lsbFilterRouteId & RouteIdMask = RouteIdBase & RouteIdMask.	RW	0x0

ddr_T_main_Probe_Filters_1_AddrBase_Low

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ddr_T_main_Probe	0xFFD12000	0xFFD12088

Offset: 0x88

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FILTERS_1_ADDRBASE_LOW RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FILTERS_1_ADDRBASE_LOW RW 0x0															

ddr_T_main_Probe_Filters_1_AddrBase_Low Fields

Bit	Name	Description	Access	Reset
31:0	FILTERS_1_ADDRBASE_LOW	Address LSB register.	RW	0x0

ddr_T_main_Probe_Filters_1_WindowSize

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ddr_T_main_Probe	0xFFD12000	0xFFD12090

Offset: 0x90

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										FILTERS_1_WINDOWSIZE RW 0x0					

ddr_T_main_Probe_Filters_1_WindowSize Fields

Bit	Name	Description	Access	Reset
5:0	FILTERS_1_WINDOWSIZE	Register WindowSize contains the encoded address mask used to filter packets. The effective Mask value is equal to ~(2max(WindowSize, packet.Len) - 1). A packet is a candidate when packet.Addr & Mask = AddrBase & Mask. This allows filteringof packets having an intersection with the AddrBase/ WindowSize burst aligned region, even if the region is smaller than the packet.	RW	0x0

ddr_T_main_Probe_Filters_1_SecurityBase

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ddr_T_main_Probe	0xFFD12000	0xFFD12094

Offset: 0x94

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													FILTERS_1_SECURITY-BASE RW 0x0		

ddr_T_main_Probe_Filters_1_SecurityBase Fields

Bit	Name	Description	Access	Reset
2:0	FILTERS_1_ SECURITYBASE	Register SecurityBase contains the security base used to filter packets.	RW	0x0

ddr_T_main_Probe_Filters_1_SecurityMask

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ddr_T_main_Probe	0xFFD12000	0xFFD12098

Offset: 0x98

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													FILTERS_1_SECURITY-MASK RW 0x0		

ddr_T_main_Probe_Filters_1_SecurityMask Fields

Bit	Name	Description	Access	Reset
2:0	FILTERS_1_ SECURITYMASK	Register SecurityMask is contains the security mask used to filter packets. A packet is a candidate when: packet.Security & Security-Mask = SecurityBase & SecurityMasks.	RW	0x0

ddr_T_main_Probe_Filters_1_Opcode

Packet Probe register Opcode is a 4-bit register that selects candidate packets based on packet opcodes (0 disables the filter):

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ddr_T_main_Probe	0xFFD12000	0xFFD1209C

Offset: 0x9C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												URGEN RW 0x0	LOCKEN RW 0x0	WREN RW 0x0	RDEN RW 0x0

ddr_T_main_Probe_Filters_1_Opcode Fields

Bit	Name	Description	Access	Reset
3	URGEN	Selects URG packets (urgency).	RW	0x0
2	LOCKEN	Selects RDX-WR, RDL, WRC and Linked sequence.	RW	0x0
1	WREN	Selects WR packets.	RW	0x0
0	RDEN	Selects RD packets.	RW	0x0

ddr_T_main_Probe_Filters_1_Status

Register Status is 2-bit register that selects candidate packets based on packet status.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ddr_T_main_Probe	0xFFD12000	0xFFD120A0

Offset: 0xA0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														RSPEN RW 0x0	REQEN RW 0x0

ddr_T_main_Probe_Filters_1_Status Fields

Bit	Name	Description	Access	Reset
1	RSPEN	Selects RSP and FAIL-CONT status packets.	RW	0x0
0	REQEN	Selects REQ status packets.	RW	0x0

ddr_T_main_Probe_Filters_1_Length

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ddr_T_main_Probe	0xFFD12000	0xFFD120A4

Offset: 0xA4

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												FILTERS_1_LENGTH RW 0x0			

ddr_T_main_Probe_Filters_1_Length Fields

Bit	Name	Description	Access	Reset
3:0	FILTERS_1_LENGTH	Register Length is 4-bit register that selects candidate packets if their number of bytes is less than or equal to 2**Length.	RW	0x0

ddr_T_main_Probe_Filters_1_Urgency

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ddr_T_main_Probe	0xFFD12000	0xFFD120A8

Offset: 0xA8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														FILTERS_1_URGENCY RW 0x0	

ddr_T_main_Probe_Filters_1_Urgency Fields

Bit	Name	Description	Access	Reset
1:0	FILTERS_1_URGENCY	Register Urgency contains the minimum urgency level used to filter packets. A packet is a candidate when its socket urgency is greater than or equal to the urgency specified in the register.	RW	0x0

ddr_T_main_Probe_Filters_2_RouteIdBase

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ddr_T_main_Probe	0xFFD12000	0xFFD120BC

Offset: 0xBC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													FILTERS_2_ROUTEIDBASE RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FILTERS_2_ROUTEIDBASE RW 0x0															

ddr_T_main_Probe_Filters_2_RouteIdBase Fields

Bit	Name	Description	Access	Reset
18:0	FILTERS_2_ROUTEIDBASE	Register RouteIdBase contains the RouteId-lsbFilterRouteId bits base used to filter packets.	RW	0x0

ddr_T_main_Probe_Filters_2_RouteIdMask

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ddr_T_main_Probe	0xFFD12000	0xFFD120C0

Offset: 0xC0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													FILTERS_2_ROUTEIDMASK RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FILTERS_2_ROUTEIDMASK RW 0x0															

ddr_T_main_Probe_Filters_2_RouteIdMask Fields

Bit	Name	Description	Access	Reset
18:0	FILTERS_2_ROUTEIDMASK	Register RouteIdMask contains the RouteId-lsbFilterRouteId mask used to filter packets. A packet is a candidate when packet.RouteId>> lsbFilterRouteId & RouteIdMask = RouteIdBase & RouteIdMask.	RW	0x0

ddr_T_main_Probe_Filters_2_AddrBase_Low

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ddr_T_main_Probe	0xFFD12000	0xFFD120C4

Offset: 0xC4

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FILTERS_2_ADDRBASE_LOW RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FILTERS_2_ADDRBASE_LOW RW 0x0															

ddr_T_main_Probe_Filters_2_AddrBase_Low Fields

Bit	Name	Description	Access	Reset
31:0	FILTERS_2_ADDRBASE_LOW	Address LSB register.	RW	0x0

ddr_T_main_Probe_Filters_2_WindowSize

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ddr_T_main_Probe	0xFFD12000	0xFFD120CC

Offset: 0xCC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										FILTERS_2_WINDOWSIZE RW 0x0					

ddr_T_main_Probe_Filters_2_WindowSize Fields

Bit	Name	Description	Access	Reset
5:0	FILTERS_2_WINDOWSIZE	Register WindowSize contains the encoded address mask used to filter packets. The effective Mask value is equal to ~(2max(WindowSize, packet.Len) - 1). A packet is a candidate when packet.Addr & Mask = AddrBase & Mask. This allows filteringof packets having an intersection with the AddrBase/ WindowSize burst aligned region, even if the region is smaller than the packet.	RW	0x0

ddr_T_main_Probe_Filters_2_SecurityBase

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ddr_T_main_Probe	0xFFD12000	0xFFD120D0

Offset: 0xD0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													FILTERS_2_SECURITY-BASE RW 0x0		

ddr_T_main_Probe_Filters_2_SecurityBase Fields

Bit	Name	Description	Access	Reset
2:0	FILTERS_2_ SECURITYBASE	Register SecurityBase contains the security base used to filter packets.	RW	0x0

ddr_T_main_Probe_Filters_2_SecurityMask

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ddr_T_main_Probe	0xFFD12000	0xFFD120D4

Offset: 0xD4

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													FILTERS_2_SECURITY-MASK RW 0x0		

ddr_T_main_Probe_Filters_2_SecurityMask Fields

Bit	Name	Description	Access	Reset
2:0	FILTERS_2_ SECURITYMASK	Register SecurityMask is contains the security mask used to filter packets. A packet is a candidate when: packet.Security & Security-Mask = SecurityBase & SecurityMasks.	RW	0x0

ddr_T_main_Probe_Filters_2_Opcode

Packet Probe register Opcode is a 4-bit register that selects candidate packets based on packet opcodes (0 disables the filter):

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ddr_T_main_Probe	0xFFD12000	0xFFD120D8

Offset: 0xD8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												URGEN RW 0x0	LOCKEN RW 0x0	WREN RW 0x0	RDEN RW 0x0

ddr_T_main_Probe_Filters_2_Opcode Fields

Bit	Name	Description	Access	Reset
3	URGEN	Selects URG packets (urgency).	RW	0x0
2	LOCKEN	Selects RDX-WR, RDL, WRC and Linked sequence.	RW	0x0
1	WREN	Selects WR packets.	RW	0x0
0	RDEN	Selects RD packets.	RW	0x0

ddr_T_main_Probe_Filters_2_Status

Register Status is 2-bit register that selects candidate packets based on packet status.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ddr_T_main_Probe	0xFFD12000	0xFFD120DC

Offset: 0xDC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														RSPEN RW 0x0	REQEN RW 0x0

ddr_T_main_Probe_Filters_2_Status Fields

Bit	Name	Description	Access	Reset
1	RSPEN	Selects RSP and FAIL-CONT status packets.	RW	0x0
0	REQEN	Selects REQ status packets.	RW	0x0

ddr_T_main_Probe_Filters_2_Length

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ddr_T_main_Probe	0xFFD12000	0xFFD120E0

Offset: 0xE0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												FILTERS_2_LENGTH RW 0x0			

ddr_T_main_Probe_Filters_2_Length Fields

Bit	Name	Description	Access	Reset
3:0	FILTERS_2_LENGTH	Register Length is 4-bit register that selects candidate packets if their number of bytes is less than or equal to 2**Length.	RW	0x0

ddr_T_main_Probe_Filters_2_Urgency

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ddr_T_main_Probe	0xFFD12000	0xFFD120E4

Offset: 0xE4

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														FILTERS_2_URGENCY RW 0x0	

ddr_T_main_Probe_Filters_2_Urgency Fields

Bit	Name	Description	Access	Reset
1:0	FILTERS_2_URGENCY	Register Urgency contains the minimum urgency level used to filter packets. A packet is a candidate when its socket urgency is greater than or equal to the urgency specified in the register.	RW	0x0

ddr_T_main_Probe_Filters_3_RouteIdBase

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ddr_T_main_Probe	0xFFD12000	0xFFD120F8

Offset: 0xF8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													FILTERS_3_ROUTEIDBASE RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FILTERS_3_ROUTEIDBASE RW 0x0															

ddr_T_main_Probe_Filters_3_RouteIdBase Fields

Bit	Name	Description	Access	Reset
18:0	FILTERS_3_ROUTEIDBASE	Register RouteIdBase contains the RouteId-lsbFilterRouteId bits base used to filter packets.	RW	0x0

ddr_T_main_Probe_Filters_3_RouteIdMask

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ddr_T_main_Probe	0xFFD12000	0xFFD120FC

Offset: 0xFC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													FILTERS_3_ROUTEIDMASK RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FILTERS_3_ROUTEIDMASK RW 0x0															

ddr_T_main_Probe_Filters_3_RouteIdMask Fields

Bit	Name	Description	Access	Reset
18:0	FILTERS_3_ROUTEIDMASK	Register RouteIdMask contains the RouteId-lsbFilterRouteId mask used to filter packets. A packet is a candidate when packet.RouteId>> lsbFilterRouteId & RouteIdMask = RouteIdBase & RouteIdMask.	RW	0x0

ddr_T_main_Probe_Filters_3_AddrBase_Low

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ddr_T_main_Probe	0xFFD12000	0xFFD12100

Offset: 0x100

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FILTERS_3_ADDRBASE_LOW RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FILTERS_3_ADDRBASE_LOW RW 0x0															

ddr_T_main_Probe_Filters_3_AddrBase_Low Fields

Bit	Name	Description	Access	Reset
31:0	FILTERS_3_ADDRBASE_LOW	Address LSB register.	RW	0x0

ddr_T_main_Probe_Filters_3_WindowSize

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ddr_T_main_Probe	0xFFD12000	0xFFD12108

Offset: 0x108

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										FILTERS_3_WINDOWSIZE RW 0x0					

ddr_T_main_Probe_Filters_3_WindowSize Fields

Bit	Name	Description	Access	Reset
5:0	FILTERS_3_WINDOWSIZE	Register WindowSize contains the encoded address mask used to filter packets. The effective Mask value is equal to ~(2max(WindowSize, packet.Len) - 1). A packet is a candidate when packet.Addr & Mask = AddrBase & Mask. This allows filteringof packets having an intersection with the AddrBase/ WindowSize burst aligned region, even if the region is smaller than the packet.	RW	0x0

ddr_T_main_Probe_Filters_3_SecurityBase

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ddr_T_main_Probe	0xFFD12000	0xFFD1210C

Offset: 0x10C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													FILTERS_3_SECURITY-BASE RW 0x0		

ddr_T_main_Probe_Filters_3_SecurityBase Fields

Bit	Name	Description	Access	Reset
2:0	FILTERS_3_ SECURITYBASE	Register SecurityBase contains the security base used to filter packets.	RW	0x0

ddr_T_main_Probe_Filters_3_SecurityMask

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ddr_T_main_Probe	0xFFD12000	0xFFD12110

Offset: 0x110

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													FILTERS_3_SECURITY-MASK RW 0x0		

ddr_T_main_Probe_Filters_3_SecurityMask Fields

Bit	Name	Description	Access	Reset
2:0	FILTERS_3_ SECURITYMASK	Register SecurityMask is contains the security mask used to filter packets. A packet is a candidate when: packet.Security & Security-Mask = SecurityBase & SecurityMasks.	RW	0x0

ddr_T_main_Probe_Filters_3_Opcode

Packet Probe register Opcode is a 4-bit register that selects candidate packets based on packet opcodes (0 disables the filter):

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ddr_T_main_Probe	0xFFD12000	0xFFD12114

Offset: 0x114

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												URGEN RW 0x0	LOCKEN RW 0x0	WREN RW 0x0	RDEN RW 0x0

ddr_T_main_Probe_Filters_3_Opcode Fields

Bit	Name	Description	Access	Reset
3	URGEN	Selects URG packets (urgency).	RW	0x0
2	LOCKEN	Selects RDX-WR, RDL, WRC and Linked sequence.	RW	0x0
1	WREN	Selects WR packets.	RW	0x0
0	RDEN	Selects RD packets.	RW	0x0

ddr_T_main_Probe_Filters_3_Status

Register Status is 2-bit register that selects candidate packets based on packet status.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ddr_T_main_Probe	0xFFD12000	0xFFD12118

Offset: 0x118

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														RSPEN RW 0x0	REQEN RW 0x0

ddr_T_main_Probe_Filters_3_Status Fields

Bit	Name	Description	Access	Reset
1	RSPEN	Selects RSP and FAIL-CONT status packets.	RW	0x0
0	REQEN	Selects REQ status packets.	RW	0x0

ddr_T_main_Probe_Filters_3_Length

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ddr_T_main_Probe	0xFFD12000	0xFFD1211C

Offset: 0x11C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												FILTERS_3_LENGTH RW 0x0			

ddr_T_main_Probe_Filters_3_Length Fields

Bit	Name	Description	Access	Reset
3:0	FILTERS_3_LENGTH	Register Length is 4-bit register that selects candidate packets if their number of bytes is less than or equal to 2**Length.	RW	0x0

ddr_T_main_Probe_Filters_3_Urgency

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ddr_T_main_Probe	0xFFD12000	0xFFD12120

Offset: 0x120

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														FILTERS_3_URGENCY RW 0x0	

ddr_T_main_Probe_Filters_3_Urgency Fields

Bit	Name	Description	Access	Reset
1:0	FILTERS_3_URGENCY	Register Urgency contains the minimum urgency level used to filter packets. A packet is a candidate when its socket urgency is greater than or equal to the urgency specified in the register.	RW	0x0

ddr_T_main_Probe_Counters_0_Src

Register CntSrc indicates the event source used to increment the counter. Unassigned values (non-existing Press level or ExtEvent index, or unimplemented Filter) are equivalent to OFF.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ddr_T_main_Probe	0xFFD12000	0xFFD12138

Offset: 0x138

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved											INTEVENT RW 0x0				

ddr_T_main_Probe_Counters_0_Src Fields

Bit	Name	Description	Access	Reset
4:0	INTEVENT	Internal packet event	RW	0x0

ddr_T_main_Probe_Counters_0_AlarmMode

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ddr_T_main_Probe	0xFFD12000	0xFFD1213C

Offset: 0x13C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														COUNTERS_0_ ALARMMODE RW 0x0	

ddr_T_main_Probe_Counters_0_AlarmMode Fields

Bit	Name	Description	Access	Reset
1:0	COUNTERS_0_ALARMMODE	Register AlarmMode is a 2-bit register that is present when parameter statisticsCounterAlarm is set to True. The register defines the statistics-alarm behavior of the counter.	RW	0x0

ddr_T_main_Probe_Counters_0_Val

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ddr_T_main_Probe	0xFFD12000	0xFFD12140

Offset: 0x140

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COUNTERS_0_VAL RO 0x0															

ddr_T_main_Probe_Counters_0_Val Fields

Bit	Name	Description	Access	Reset
15:0	COUNTERS_0_VAL	Register Val is a read-only register that is always present. The register containsthe statistics counter value either pending StatAlarm output, or when statisticscollection is suspended subsequent to triggers or signal statSuspend.	RO	0x0

ddr_T_main_Probe_Counters_1_Src

Register CntSrc indicates the event source used to increment the counter. Unassigned values (non-existing Press level or ExtEvent index, or unimplemented Filter) are equivalent to OFF.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ddr_T_main_Probe	0xFFD12000	0xFFD1214C

Offset: 0x14C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved											INTEVENT RW 0x0				

ddr_T_main_Probe_Counters_1_Src Fields

Bit	Name	Description	Access	Reset
4:0	INTEVENT	Internal packet event	RW	0x0

ddr_T_main_Probe_Counters_1_AlarmMode

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ddr_T_main_Probe	0xFFD12000	0xFFD12150

Offset: 0x150

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														COUNTERS_1_ALARMMODE RW 0x0	

ddr_T_main_Probe_Counters_1_AlarmMode Fields

Bit	Name	Description	Access	Reset
1:0	COUNTERS_1_ALARMMODE	Register AlarmMode is a 2-bit register that is present when parameter statisticsCounterAlarm is set to True. The register defines the statistics-alarm behavior of the counter.	RW	0x0

ddr_T_main_Probe_Counters_1_Val

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ddr_T_main_Probe	0xFFD12000	0xFFD12154

Offset: 0x154

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COUNTERS_1_VAL RO 0x0															

ddr_T_main_Probe_Counters_1_Val Fields

Bit	Name	Description	Access	Reset
15:0	COUNTERS_1_VAL	Register Val is a read-only register that is always present. The register containsthe statistics counter value either pending StatAlarm output, or when statisticscollection is suspended subsequent to triggers or signal statSuspend.	RO	0x0

noc_mpu_m0_ddr_T_main_Scheduler Address Map

Module Instance	Base Address	End Address
i_noc_mpu_m0_ddr_T_main_Scheduler	0xFFD12400	0xFFD12FFF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
ddr_T_main_Scheduler_Id_CoreId on page 7-304	0x0	32	RO	0x7242E202	
ddr_T_main_Scheduler_Id_RevisionId on page 7-305	0x4	32	RO	0x129FF00	
ddr_T_main_Scheduler_DdrConf on page 7-306	0x8	32	RW	0x0	ddr configuration definition.
ddr_T_main_Scheduler_DdrTiming on page 7-306	0xC	32	RW	0xAC2A14DC	ddr timing definition.
ddr_T_main_Scheduler_DdrMode on page 7-308	0x10	32	RW	0x0	ddr mode definition.
ddr_T_main_Scheduler_ReadLatency on page 7-309	0x14	32	RW	0x13	
ddr_T_main_Scheduler_Activate on page 7-310	0x38	32	RW	0x4D2	timing values concerning Activate commands, in Generic clock unit.
ddr_T_main_Scheduler_DevToDev on page 7-311	0x3C	32	RW	0x15	timing values concerning device to device data bus ownership change, in Generic clock unit.

noc_mpu_m0_ddr_T_main_Scheduler Summary

Base Address: 0xFFD12400

Register Address Offset	Bit Fields															
i_noc_mpu_m0_ddr_T_main_Scheduler																
ddr_T_main_Scheduler_Id_CoreId 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	CORECHECKSUM RO 0x7242E2															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CORECHECKSUM RO 0x7242E2								CORETYPEID RO 0x2							
ddr_T_main_Scheduler_Id_RevisionId 0x4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	FLEXNOCID RO 0x129FF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FLEXNOCID RO 0x129FF								USERID RO 0x0							
ddr_T_main_Scheduler_DdrConf 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved											DDRCONF RW 0x0				
ddr_T_main_Scheduler_DdrTiming 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	BWRATIO RW 0x1	WRTORD RW 0xB					RDTOWR RW 0x1					BURSTLEN RW 0x2			WRTOMISS RW 0x21	
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	WRTOMISS RW 0x21				RDTOMISS RW 0x13						ACTTOACT RW 0x1C					
ddr_T_main_Scheduler_DdrMode 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														BWRATIOEXTENDED RW 0x0	AUTOPRE-CHARGE RW 0x0

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ddr_T_main_Scheduler_ReadLatency 0x14	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								READLATENCY RW 0x13							
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ddr_T_main_Scheduler_Activate 0x38	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved					FAWBANK RW 0x1	FAW RW 0xD						RRD RW 0x2			
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ddr_T_main_Scheduler_DevToDev 0x3C	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved										BUSWRTORD RW 0x1		BUSRDTOWR RW 0x1		BUSRDTORD RW 0x1	

ddr_T_main_Scheduler_Id_CoreId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ddr_T_main_Scheduler	0xFFD12400	0xFFD12400

Offset: 0x0

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CORECHECKSUM RO 0x7242E2															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CORECHECKSUM RO 0x7242E2								CORETYPEID RO 0x2							

ddr_T_main_Scheduler_Id_CoreId Fields

Bit	Name	Description	Access	Reset
31:8	CORECHECKSUM	Field containing a checksum of the parameters of the IP.	RO	0x7242E2
7:0	CORETYPEID	Field identifying the type of IP.	RO	0x2

ddr_T_main_Scheduler_Id_RevisionId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ddr_T_main_Scheduler	0xFFD12400	0xFFD12404

Offset: 0x4

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLEXNOCID RO 0x129FF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLEXNOCID RO 0x129FF								USERID RO 0x0							

ddr_T_main_Scheduler_Id_RevisionId Fields

Bit	Name	Description	Access	Reset
31:8	FLEXNOCID	Field containing the build revision of the software used to generate the IP HDL code.	RO	0x129FF
7:0	USERID	Field containing a user defined value, not used anywhere inside the IP itself.	RO	0x0

ddr_T_main_Scheduler_DdrConf

ddr configuration definition.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ddr_T_main_Scheduler	0xFFD12400	0xFFD12408

Offset: 0x8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved											DDRCONF RW 0x0				

ddr_T_main_Scheduler_DdrConf Fields

Bit	Name	Description	Access	Reset
4:0	DDRCONF	Selection of a configuration of mappings of address bits to memory device, bank, row, and column. For more information, refer to the SoC-specific DDR Conf documentation.	RW	0x0

ddr_T_main_Scheduler_DdrTiming

ddr timing definition.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ddr_T_main_Scheduler	0xFFD12400	0xFFD1240C

Offset: 0xC

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BWRATIO RW 0x1	WRTORD RW 0xB					RDTOWR RW 0x1					BURSTLEN RW 0x2			WRTOMISS RW 0x21	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WRTOMISS RW 0x21				RDTOMISS RW 0x13						ACTTOACT RW 0x1C					

ddr_T_main_Scheduler_DdrTiming Fields

Bit	Name	Description	Access	Reset
31	BWRATIO	When set to zero, one DRAM clock cycle (two DDR transfers) is used to transfer each word of data. When set to one, two DRAM clock cycles (four DDR transfers) are used to transfer each word of data. This is applicable when half of a DRAM data bus width is used.	RW	0x1
30:26	WRTORD	The minimum number of scheduler clock cycles between the last DRAM Write command and a Read command (WL x tCkD + tWTR). tCkD is the DRAM clock period.	RW	0xB
25:21	RDTOWR	The minimum number of scheduler clock cycles between the last DRAM Read command and a Write command (DDR3: (RL – WL + 2) x tCkD). tCkD is the DRAM clock period.	RW	0x1
20:18	BURSTLEN	The DRAM burst duration on the DRAM data bus in scheduler clock cycles. Also equal to scheduler clock cycles between two DRAM commands (BL / 2 x tCkD). tCkD is the DRAM clock period.	RW	0x2

Bit	Name	Description	Access	Reset
17:12	WRTOMISS	The minimum number of scheduler clock cycles between the last DRAM Write command and a new Read or Write command in another page of the same bank (WL x tCkD + tWR + tRP + tRCD). tCkD is the DRAM clock period.	RW	0x21
11:6	RDTOMISS	The minimum number of scheduler clock cycles between the last DRAM Read command and a new Read or Write command in another page of the same bank (tRTP + tRP + tRCD – BL x tCkD / 2). tCkD is the DRAM clock period.	RW	0x13
5:0	ACTTOACT	The minimum number of scheduler clock cycles between two consecutive DRAM Activate commands on the same bank (tRC/ tCkG). tCkG is the clock period of the SoC DRAM scheduler.	RW	0x1C

ddr_T_main_Scheduler_DdrMode

ddr mode definition.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ddr_T_main_Scheduler	0xFFD12400	0xFFD12410

Offset: 0x10

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														BWRATIOEXTENDED RW 0x0	AUTOPRE-CHARGE RW 0x0

ddr_T_main_Scheduler_DdrMode Fields

Bit	Name	Description	Access	Reset
1	BWRATIOEXTENDED	When set to one, four DRAM clock cycles (8 DDR transfers) are used to transfer each word of data. When set to one, the BwRatio field of the DdrTiming register must be set to zero.	RW	0x0
0	AUTOPRECHARGE	When set to one, pages are automatically closed after each access, when set to zero, pages are left opened until an access in a different page occurs.	RW	0x0

ddr_T_main_Scheduler_ReadLatency

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ddr_T_main_Scheduler	0xFFD12400	0xFFD12414

Offset: 0x14

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								READLATENCY RW 0x13							

ddr_T_main_Scheduler_ReadLatency Fields

Bit	Name	Description	Access	Reset
7:0	READLATENCY	The DRAM type-specific number of cycles from a scheduler request to a protocol controller response. This is a fixed value depending on the type of DRAM memory. For more information, refer to the SoC-specific memory controller documentation.	RW	0x13

ddr_T_main_Scheduler_Activate

timing values concerning Activate commands, in Generic clock unit.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ddr_T_main_Scheduler	0xFFD12400	0xFFD12438

Offset: 0x38

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					FAWBANK RW 0x1	FAW RW 0xD						RRD RW 0x2			

ddr_T_main_Scheduler_Activate Fields

Bit	Name	Description	Access	Reset
10	FAWBANK	The number of Banks of a given device involved in the FAW period. Set to zero for 2-bank memories (WideIO). Set to one for memories with 4 banks or more (DDR).	RW	0x1
9:4	FAW	The number of cycles for the four bank activate (FAW) period (tFAW).	RW	0xD
3:0	RRD	The number of cycles between two consecutive Activate commands on different Banks of the same device (tRRD).	RW	0x2

ddr_T_main_Scheduler_DevToDev

timing values concerning device to device data bus ownership change, in Generic clock unit.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ddr_T_main_Scheduler	0xFFD12400	0xFFD1243C

Offset: 0x3C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										BUSWRTORD RW 0x1		BUSRDTOWR RW 0x1		BUSRDTORD RW 0x1	

ddr_T_main_Scheduler_DevToDev Fields

Bit	Name	Description	Access	Reset
5:4	BUSWRTORD	The number of cycles between the last write data to a device and the first read data of another device of a memory array with multiple ranks (2 x tCkD). tCkD is the DRAM clock period.	RW	0x1
3:2	BUSRDTOWR	The number of cycles between the last read data of a device and the first write data to another device of a memory array with multiple ranks (2 x tCkD). tCkD is the DRAM clock period.	RW	0x1
1:0	BUSRDTORD	The number of cycles between the last read data of a device and the first read data of another device of a memory array with multiple ranks (tCkD). tCkD is the DRAM clock period.	RW	0x1

noc_fw_l4_per_l4_per_scr Address Map

Module Instance	Base Address	End Address
noc_fw_l4_per_l4_per_scr	0xFFD13000	0xFFD130FF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
nand_register on page 7-321	0x0	32	RW	0x0	Per-Master Security bit for nand register
nand_data on page 7-322	0x4	32	RW	0x0	Per-Master Security bit for nand_data
qspi_data on page 7-324	0x8	32	RW	0x0	Per-Master Security bit for qspi_data
usb0_register on page 7-325	0xC	32	RW	0x0	Per-Master Security bit for usb0_register
usb1_register on page 7-326	0x10	32	RW	0x0	Per-Master Security bit for usb1_register
dma_nonsecure on page 7-328	0x14	32	RO	0x1	Per-Master Security bit for dma_nonsecure
dma_secure on page 7-328	0x18	32	RO	0x0	Per-Master Security bit for dma_secure
spi_master0 on page 7-329	0x1C	32	RW	0x0	Per-Master Security bit for spi_master0
spi_master1 on page 7-330	0x20	32	RW	0x0	Per-Master Security bit for spi_master1
spi_slave0 on page 7-331	0x24	32	RW	0x0	Per-Master Security bit for spi_slave0
spi_slave1 on page 7-333	0x28	32	RW	0x0	Per-Master Security bit for spi_slave1
emac0 on page 7-334	0x2C	32	RW	0x0	Per-Master Security bit for emac0
emac1 on page 7-335	0x30	32	RW	0x0	Per-Master Security bit for emac1
emac2 on page 7-337	0x34	32	RW	0x0	Per-Master Security bit for emac2
emac3 on page 7-338	0x38	32	RO	0x0	Per-Master Security bit for emac3
qspi on page 7-339	0x3C	32	RW	0x0	Per-Master Security bit for qspi

Register	Offset	Width	Access	Reset Value	Description
sdmmc on page 7-340	0x40	32	RW	0x0	Per-Master Security bit for sdmmc
gpio0 on page 7-341	0x44	32	RW	0x0	Per-Master Security bit for gpio0
gpio1 on page 7-343	0x48	32	RW	0x0	Per-Master Security bit for gpio1
gpio2 on page 7-344	0x4C	32	RW	0x0	Per-Master Security bit for gpio2
i2c0 on page 7-345	0x50	32	RW	0x0	Per-Master Security bit for i2c0
i2c1 on page 7-347	0x54	32	RW	0x0	Per-Master Security bit for i2c1
i2c2 on page 7-348	0x58	32	RW	0x0	Per-Master Security bit for i2c2
i2c3 on page 7-349	0x5C	32	RW	0x0	Per-Master Security bit for i2c3
i2c4 on page 7-351	0x60	32	RW	0x0	Per-Master Security bit for i2c4
sp_timer0 on page 7-352	0x64	32	RW	0x0	Per-Master Security bit for sp_timer0
sp_timer1 on page 7-353	0x68	32	RW	0x0	Per-Master Security bit for sp_timer1
uart0 on page 7-355	0x6C	32	RW	0x0	Per-Master Security bit for uart0
uart1 on page 7-356	0x70	32	RW	0x0	Per-Master Security bit for uart1

noc_fw_l4_per_l4_per_scr Summary

Base Address: 0xFFD13000

Register Address Offset	Bit Fields															
noc_fw_l4_per_l4_per_scr																
nand_register 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0
nand_data 0x4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0
qspi_data 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0
usb0_register 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0

Register Address Offset	Bit Fields															
usb1_register 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0
dma_nonsecure 0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															
dma_secure 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															
spi_master0 0x1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0
spi_master1 0x20	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0

Register Address Offset	Bit Fields															
spi_slave0 0x24	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0
spi_slave1 0x28	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0
emac0 0x2C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0
emac1 0x30	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0

Register Address Offset	Bit Fields															
emac2 0x34	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0
emac3 0x38	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															
qspi 0x3C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0
sdmmc 0x40	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0
gpio0 0x44	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gpio1 0x48	Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gpio2 0x4C	Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
i2c0 0x50	Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
i2c1 0x54	Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
i2c2 0x58	Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
i2c3 0x5C	Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
i2c4 0x60	Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
sp_timer0 0x64	Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0

Register Address Offset	Bit Fields															
sp_timer1 0x68	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0
uart0 0x6C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0
uart1 0x70	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0

nand_register

Per-Master Security bit for nand register

Module Instance	Base Address	Register Address
noc_fw_l4_per_l4_per_scr	0xFFD13000	0xFFD13000

Offset: 0x0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0

nand_register Fields

Bit	Name	Description	Access	Reset
24	ahb_ap	Security bit configuration for transactions from ahb_ap to nand_ register. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
16	fpga2soc	Security bit configuration for transactions from fpga2soc to nand_ register. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
8	dma	Security bit configuration for transactions from dma to nand_ register. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
0	mpu_m0	Security bit configuration for transactions from mpu_m0 to nand_ register. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

nand_data

Per-Master Security bit for nand_data

Module Instance	Base Address	Register Address
noc_fw_l4_per_l4_per_scr	0xFFD13000	0xFFD13004

Offset: 0x4

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0

nand_data Fields

Bit	Name	Description	Access	Reset
24	ahb_ap	Security bit configuration for transactions from ahb_ap to nand_data. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
16	fpga2soc	Security bit configuration for transactions from fpga2soc to nand_data. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
8	dma	Security bit configuration for transactions from dma to nand_data. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

Bit	Name	Description	Access	Reset
0	mpu_m0	Security bit configuration for transactions from mpu_m0 to nand_data. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

qspi_data

Per-Master Security bit for qspi_data

Module Instance	Base Address	Register Address
noc_fw_l4_per_l4_per_scr	0xFFD13000	0xFFD13008

Offset: 0x8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0

qspi_data Fields

Bit	Name	Description	Access	Reset
24	ahb_ap	Security bit configuration for transactions from ahb_ap to qspi_data. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

Bit	Name	Description	Access	Reset
16	fpga2soc	Security bit configuration for transactions from fpga2soc to qspi_data. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
8	dma	Security bit configuration for transactions from dma to qspi_data. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
0	mpu_m0	Security bit configuration for transactions from mpu_m0 to qspi_data. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

usb0_register

Per-Master Security bit for usb0_register

Module Instance	Base Address	Register Address
noc_fw_l4_per_l4_per_scr	0xFFD13000	0xFFD1300C

Offset: 0xC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0

usb0_register Fields

Bit	Name	Description	Access	Reset
24	ahb_ap	Security bit configuration for transactions from ahb_ap to usb0_ register. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
16	fpga2soc	Security bit configuration for transactions from fpga2soc to usb0_ register. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
8	dma	Security bit configuration for transactions from dma to usb0_ register. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
0	mpu_m0	Security bit configuration for transactions from mpu_m0 to usb0_ register. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

usb1_register

Per-Master Security bit for usb1_register

Module Instance	Base Address	Register Address
noc_fw_l4_per_l4_per_scr	0xFFD13000	0xFFD13010

Offset: 0x10

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0

usb1_register Fields

Bit	Name	Description	Access	Reset
24	ahb_ap	Security bit configuration for transactions from ahb_ap to usb1_register. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
16	fpga2soc	Security bit configuration for transactions from fpga2soc to usb1_register. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
8	dma	Security bit configuration for transactions from dma to usb1_register. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

Bit	Name	Description	Access	Reset
0	mpu_m0	Security bit configuration for transactions from mpu_m0 to usb1_ register. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

dma_nonsecure

Per-Master Security bit for dma_nonsecure

Module Instance	Base Address	Register Address
noc_fw_l4_per_l4_per_scr	0xFFD13000	0xFFD13014

Offset: 0x14

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															

dma_nonsecure Fields

Bit	Name	Description	Access	Reset
31:0	Reserved		RO	0x1

dma_secure

Per-Master Security bit for dma_secure

Module Instance	Base Address	Register Address
noc_fw_l4_per_l4_per_scr	0xFFD13000	0xFFD13018

Offset: 0x18

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															

dma_secure Fields

Bit	Name	Description	Access	Reset
31:0	Reserved		RO	0x0

spi_master0

Per-Master Security bit for spi_master0

Module Instance	Base Address	Register Address
noc_fw_l4_per_l4_per_scr	0xFFD13000	0xFFD1301C

Offset: 0x1C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0

spi_master0 Fields

Bit	Name	Description	Access	Reset
24	ahb_ap	Security bit configuration for transactions from ahb_ap to spi_master0. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
16	fpga2soc	Security bit configuration for transactions from fpga2soc to spi_master0. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
8	dma	Security bit configuration for transactions from dma to spi_master0. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
0	mpu_m0	Security bit configuration for transactions from mpu_m0 to spi_master0. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

spi_master1

Per-Master Security bit for spi_master1

Module Instance	Base Address	Register Address
noc_fw_l4_per_l4_per_scr	0xFFD13000	0xFFD13020

Offset: 0x20

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0

spi_master1 Fields

Bit	Name	Description	Access	Reset
24	ahb_ap	Security bit configuration for transactions from ahb_ap to spi_master1. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
16	fpga2soc	Security bit configuration for transactions from fpga2soc to spi_master1. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
8	dma	Security bit configuration for transactions from dma to spi_master1. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
0	mpu_m0	Security bit configuration for transactions from mpu_m0 to spi_master1. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

spi_slave0

Per-Master Security bit for spi_slave0

Module Instance	Base Address	Register Address
noc_fw_l4_per_l4_per_scr	0xFFD13000	0xFFD13024

Offset: 0x24

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0

spi_slave0 Fields

Bit	Name	Description	Access	Reset
24	ahb_ap	Security bit configuration for transactions from ahb_ap to spi_slave0. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
16	fpga2soc	Security bit configuration for transactions from fpga2soc to spi_slave0. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
8	dma	Security bit configuration for transactions from dma to spi_slave0. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

Bit	Name	Description	Access	Reset
0	mpu_m0	Security bit configuration for transactions from mpu_m0 to spi_slave0. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

spi_slave1

Per-Master Security bit for spi_slave1

Module Instance	Base Address	Register Address
noc_fw_l4_per_l4_per_scr	0xFFD13000	0xFFD13028

Offset: 0x28

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0

spi_slave1 Fields

Bit	Name	Description	Access	Reset
24	ahb_ap	Security bit configuration for transactions from ahb_ap to spi_slave1. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

Bit	Name	Description	Access	Reset
16	fpga2soc	Security bit configuration for transactions from fpga2soc to spi_slave1. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
8	dma	Security bit configuration for transactions from dma to spi_slave1. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
0	mpu_m0	Security bit configuration for transactions from mpu_m0 to spi_slave1. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

emac0

Per-Master Security bit for emac0

Module Instance	Base Address	Register Address
noc_fw_l4_per_l4_per_scr	0xFFD13000	0xFFD1302C

Offset: 0x2C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0

emac0 Fields

Bit	Name	Description	Access	Reset
24	ahb_ap	Security bit configuration for transactions from ahb_ap to emac0. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
16	fpga2soc	Security bit configuration for transactions from fpga2soc to emac0. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
8	dma	Security bit configuration for transactions from dma to emac0. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
0	mpu_m0	Security bit configuration for transactions from mpu_m0 to emac0. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

emac1

Per-Master Security bit for emac1

Module Instance	Base Address	Register Address
noc_fw_l4_per_l4_per_scr	0xFFD13000	0xFFD13030

Offset: 0x30

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0

emac1 Fields

Bit	Name	Description	Access	Reset
24	ahb_ap	Security bit configuration for transactions from ahb_ap to emac1. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
16	fpga2soc	Security bit configuration for transactions from fpga2soc to emac1. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
8	dma	Security bit configuration for transactions from dma to emac1. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

Bit	Name	Description	Access	Reset
0	mpu_m0	Security bit configuration for transactions from mpu_m0 to emac1. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

emac2

Per-Master Security bit for emac2

Module Instance	Base Address	Register Address
noc_fw_l4_per_l4_per_scr	0xFFD13000	0xFFD13034

Offset: 0x34

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0

emac2 Fields

Bit	Name	Description	Access	Reset
24	ahb_ap	Security bit configuration for transactions from ahb_ap to emac2. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

Bit	Name	Description	Access	Reset
16	fpga2soc	Security bit configuration for transactions from fpga2soc to emac2. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
8	dma	Security bit configuration for transactions from dma to emac2. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
0	mpu_m0	Security bit configuration for transactions from mpu_m0 to emac2. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

emac3

Per-Master Security bit for emac3

Module Instance	Base Address	Register Address
noc_fw_l4_per_l4_per_scr	0xFFD13000	0xFFD13038

Offset: 0x38

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															

emac3 Fields

Bit	Name	Description	Access	Reset
31:0	Reserved		RO	0x0

qspi

Per-Master Security bit for qspi

Module Instance	Base Address	Register Address
noc_fw_l4_per_l4_per_scr	0xFFD13000	0xFFD1303C

Offset: 0x3C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0

qspi Fields

Bit	Name	Description	Access	Reset
24	ahb_ap	Security bit configuration for transactions from ahb_ap to qspi. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

Bit	Name	Description	Access	Reset
16	fpga2soc	Security bit configuration for transactions from fpga2soc to qspi. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
8	dma	Security bit configuration for transactions from dma to qspi. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
0	mpu_m0	Security bit configuration for transactions from mpu_m0 to qspi. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

sdmmc

Per-Master Security bit for sdmmc

Module Instance	Base Address	Register Address
noc_fw_l4_per_l4_per_scr	0xFFD13000	0xFFD13040

Offset: 0x40

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0

sdmmc Fields

Bit	Name	Description	Access	Reset
24	ahb_ap	Security bit configuration for transactions from ahb_ap to sdmmc. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
16	fpga2soc	Security bit configuration for transactions from fpga2soc to sdmmc. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
8	dma	Security bit configuration for transactions from dma to sdmmc. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
0	mpu_m0	Security bit configuration for transactions from mpu_m0 to sdmmc. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

gpio0

Per-Master Security bit for gpio0

Module Instance	Base Address	Register Address
noc_fw_l4_per_l4_per_scr	0xFFD13000	0xFFD13044

Offset: 0x44

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0

gpio0 Fields

Bit	Name	Description	Access	Reset
24	ahb_ap	Security bit configuration for transactions from ahb_ap to gpio0. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
16	fpga2soc	Security bit configuration for transactions from fpga2soc to gpio0. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
8	dma	Security bit configuration for transactions from dma to gpio0. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

Bit	Name	Description	Access	Reset
0	mpu_m0	Security bit configuration for transactions from mpu_m0 to gpio0. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

gpio1

Per-Master Security bit for gpio1

Module Instance	Base Address	Register Address
noc_fw_l4_per_l4_per_scr	0xFFD13000	0xFFD13048

Offset: 0x48

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0

gpio1 Fields

Bit	Name	Description	Access	Reset
24	ahb_ap	Security bit configuration for transactions from ahb_ap to gpio1. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

Bit	Name	Description	Access	Reset
16	fpga2soc	Security bit configuration for transactions from fpga2soc to gpio1. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
8	dma	Security bit configuration for transactions from dma to gpio1. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
0	mpu_m0	Security bit configuration for transactions from mpu_m0 to gpio1. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

gpio2

Per-Master Security bit for gpio2

Module Instance	Base Address	Register Address
noc_fw_l4_per_l4_per_scr	0xFFD13000	0xFFD1304C

Offset: 0x4C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0

gpio2 Fields

Bit	Name	Description	Access	Reset
24	ahb_ap	Security bit configuration for transactions from ahb_ap to gpio2. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
16	fpga2soc	Security bit configuration for transactions from fpga2soc to gpio2. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
8	dma	Security bit configuration for transactions from dma to gpio2. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
0	mpu_m0	Security bit configuration for transactions from mpu_m0 to gpio2. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

i2c0

Per-Master Security bit for i2c0

Module Instance	Base Address	Register Address
noc_fw_l4_per_l4_per_scr	0xFFD13000	0xFFD13050

Offset: 0x50

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0

i2c0 Fields

Bit	Name	Description	Access	Reset
24	ahb_ap	Security bit configuration for transactions from ahb_ap to i2c0. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
16	fpga2soc	Security bit configuration for transactions from fpga2soc to i2c0. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
8	dma	Security bit configuration for transactions from dma to i2c0. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

Bit	Name	Description	Access	Reset
0	mpu_m0	Security bit configuration for transactions from mpu_m0 to i2c0. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

i2c1

Per-Master Security bit for i2c1

Module Instance	Base Address	Register Address
noc_fw_l4_per_l4_per_scr	0xFFD13000	0xFFD13054

Offset: 0x54

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0

i2c1 Fields

Bit	Name	Description	Access	Reset
24	ahb_ap	Security bit configuration for transactions from ahb_ap to i2c1. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

Bit	Name	Description	Access	Reset
16	fpga2soc	Security bit configuration for transactions from fpga2soc to i2c1. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
8	dma	Security bit configuration for transactions from dma to i2c1. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
0	mpu_m0	Security bit configuration for transactions from mpu_m0 to i2c1. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

i2c2

Per-Master Security bit for i2c2

Module Instance	Base Address	Register Address
noc_fw_l4_per_l4_per_scr	0xFFD13000	0xFFD13058

Offset: 0x58

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0

i2c2 Fields

Bit	Name	Description	Access	Reset
24	ahb_ap	Security bit configuration for transactions from ahb_ap to i2c2. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
16	fpga2soc	Security bit configuration for transactions from fpga2soc to i2c2. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
8	dma	Security bit configuration for transactions from dma to i2c2. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
0	mpu_m0	Security bit configuration for transactions from mpu_m0 to i2c2. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

i2c3

Per-Master Security bit for i2c3

Module Instance	Base Address	Register Address
noc_fw_l4_per_l4_per_scr	0xFFD13000	0xFFD1305C

Offset: 0x5C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0

i2c3 Fields

Bit	Name	Description	Access	Reset
24	ahb_ap	Security bit configuration for transactions from ahb_ap to i2c3. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
16	fpga2soc	Security bit configuration for transactions from fpga2soc to i2c3. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
8	dma	Security bit configuration for transactions from dma to i2c3. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

Bit	Name	Description	Access	Reset
0	mpu_m0	Security bit configuration for transactions from mpu_m0 to i2c3. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

i2c4

Per-Master Security bit for i2c4

Module Instance	Base Address	Register Address
noc_fw_l4_per_l4_per_scr	0xFFD13000	0xFFD13060

Offset: 0x60

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0

i2c4 Fields

Bit	Name	Description	Access	Reset
24	ahb_ap	Security bit configuration for transactions from ahb_ap to. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

Bit	Name	Description	Access	Reset
16	fpga2soc	Security bit configuration for transactions from fpga2soc to i2c4. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
8	dma	Security bit configuration for transactions from dma to i2c4. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
0	mpu_m0	Security bit configuration for transactions from mpu_m0 to i2c4. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

sp_timer0

Per-Master Security bit for sp_timer0

Module Instance	Base Address	Register Address
noc_fw_l4_per_l4_per_scr	0xFFD13000	0xFFD13064

Offset: 0x64

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0

sp_timer0 Fields

Bit	Name	Description	Access	Reset
24	ahb_ap	Security bit configuration for transactions from ahb_ap to sp_timer0. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
16	fpga2soc	Security bit configuration for transactions from fpga2soc to sp_timer0. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
8	dma	Security bit configuration for transactions from dma to sp_timer0. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
0	mpu_m0	Security bit configuration for transactions from mpu_m0 to sp_timer0. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

sp_timer1

Per-Master Security bit for sp_timer1

Module Instance	Base Address	Register Address
noc_fw_l4_per_l4_per_scr	0xFFD13000	0xFFD13068

Offset: 0x68

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0

sp_timer1 Fields

Bit	Name	Description	Access	Reset
24	ahb_ap	Security bit configuration for transactions from ahb_ap to sp_timer1. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
16	fpga2soc	Security bit configuration for transactions from fpga2soc to sp_timer1. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
8	dma	Security bit configuration for transactions from dma to sp_timer1. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

Bit	Name	Description	Access	Reset
0	mpu_m0	Security bit configuration for transactions from mpu_m0 to sp_timer1. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

uart0

Per-Master Security bit for uart0

Module Instance	Base Address	Register Address
noc_fw_l4_per_l4_per_scr	0xFFD13000	0xFFD1306C

Offset: 0x6C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0

uart0 Fields

Bit	Name	Description	Access	Reset
24	ahb_ap	Security bit configuration for transactions from ahb_ap to uart0. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

Bit	Name	Description	Access	Reset
16	fpga2soc	Security bit configuration for transactions from fpga2soc to uart0. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
8	dma	Security bit configuration for transactions from dma to uart0. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
0	mpu_m0	Security bit configuration for transactions from mpu_m0 to uart0. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

uart1

Per-Master Security bit for uart1

Module Instance	Base Address	Register Address
noc_fw_l4_per_l4_per_scr	0xFFD13000	0xFFD13070

Offset: 0x70

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0

uart1 Fields

Bit	Name	Description	Access	Reset
24	ahb_ap	Security bit configuration for transactions from ahb_ap to uart1. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
16	fpga2soc	Security bit configuration for transactions from fpga2soc to uart1. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
8	dma	Security bit configuration for transactions from dma to uart1. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
0	mpu_m0	Security bit configuration for transactions from mpu_m0 to uart1. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

noc_fw_l4_sys_l4_sys_scr Address Map

Module Instance	Base Address	End Address
noc_fw_l4_sys_l4_sys_scr	0xFFD13100	0xFFD131FF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
dma_ecc on page 7-368	0x8	32	RW	0x0	Per-Master Security bit for dma_ecc
emac0rx_ecc on page 7-369	0xC	32	RW	0x0	Per-Master Security bit for emac0rx_ecc
emac0tx_ecc on page 7-370	0x10	32	RW	0x0	Per-Master Security bit for emac0tx_ecc
emac1rx_ecc on page 7-371	0x14	32	RW	0x0	Per-Master Security bit for emac1rx_ecc
emac1tx_ecc on page 7-373	0x18	32	RW	0x0	Per-Master Security bit for emac1tx_ecc
emac2rx_ecc on page 7-374	0x1C	32	RW	0x0	Per-Master Security bit for emac2rx_ecc
emac2tx_ecc on page 7-375	0x20	32	RW	0x0	Per-Master Security bit for emac2tx_ecc
emac3rx_ecc on page 7-376	0x24	32	RO	0x0	Per-Master Security bit for emac3rx_ecc
emac3tx_ecc on page 7-377	0x28	32	RO	0x0	Per-Master Security bit for emac3tx_ecc
nand_ecc on page 7-377	0x2C	32	RW	0x0	Per-Master Security bit for nand_ecc
nand_read_ecc on page 7-378	0x30	32	RW	0x0	Per-Master Security bit for nand_read_ecc
nand_write_ecc on page 7-380	0x34	32	RW	0x0	Per-Master Security bit for nand_write_ecc
onchipram_ecc on page 7-381	0x38	32	RW	0x0	Per-Master Security bit for onchipram_ecc

Register	Offset	Width	Access	Reset Value	Description
qspi_ecc on page 7-382	0x3C	32	RW	0x0	Per-Master Security bit for qspi_ecc
sdmmc_ecc on page 7-383	0x40	32	RW	0x0	Per-Master Security bit for sdmmc_ecc
usb0_ecc on page 7-384	0x44	32	RW	0x0	Per-Master Security bit for usb0_ecc
usb1_ecc on page 7-385	0x48	32	RW	0x0	Per-Master Security bit for usb1_ecc
clock_manager on page 7-386	0x4C	32	RW	0x0	Per-Master Security bit for clock_manager
fpga_manager_register on page 7-388	0x50	32	RW	0x0	Per-Master Security bit for fpga_manager
pin_mux_register on page 7-389	0x54	32	RW	0x0	Per-Master Security bit for pin_mux_ register
reset_manager on page 7-390	0x58	32	RW	0x0	Per-Master Security bit for reset_manager
system_manager on page 7-391	0x5C	32	RW	0x0	Per-Master Security bit for system_manager
osc0_timer on page 7-393	0x60	32	RW	0x0	Per-Master Security bit for osc0_timer
osc1_timer on page 7-394	0x64	32	RW	0x0	Per-Master Security bit for osc1_timer
watchdog0 on page 7-395	0x68	32	RW	0x0	Per-Master Security bit for watchdog0
watchdog1 on page 7-397	0x6C	32	RW	0x0	Per-Master Security bit for watchdog1
dap on page 7-398	0x70	32	RW	0x0	Per-Master Security bit for dap
fpga_manager_streaming on page 7-400	0x74	32	RW	0x0	Per-Master Security bit for fpga_manager_ streaming

Register	Offset	Width	Access	Reset Value	Description
security_manager_streaming on page 7-401	0x78	32	RW	0x0	Per-Master Security bit for security_manager_streaming
hmc_register on page 7-402	0x7C	32	RW	0x0	Per-Master Security bit for hmc_register
hmc_adaptor_register on page 7-403	0x80	32	RW	0x0	Per-Master Security bit for hmc_adaptor_register
l3_interconnect_register on page 7-405	0x84	32	RW	0x0	Per-Master Security bit for ddr_scheduler_register
ddr_scheduler_register on page 7-406	0x88	32	RW	0x0	Per-Master Security bit for ddr_scheduler_register
l4_interconnect_firewall_csr on page 7-407	0x8C	32	RW	0x0	Per-Master Security bit for noc_firewall_scr
l4_interconnect_probes_csr on page 7-408	0x90	32	RW	0x0	Per-Master Security bit for noc_probes_register
l4_qos_csr on page 7-410	0x94	32	RW	0x0	Per-Master Security bit for noc_probes_register

noc_fw_l4_sys_l4_sys_scr Summary

Base Address: 0xFFD13100

Register Address Offset	Bit Fields
noc_fw_l4_sys_l4_sys_scr	

Register Address Offset	Bit Fields															
dma_ecc 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															mpu_m0 RW 0x0
emac0rx_ecc 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															mpu_m0 RW 0x0
emac0tx_ecc 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															mpu_m0 RW 0x0
emac1rx_ecc 0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															mpu_m0 RW 0x0
emac1tx_ecc 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															mpu_m0 RW 0x0

Register Address Offset	Bit Fields															
emac2rx_ecc 0x1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															mpu_m0 RW 0x0
emac2tx_ecc 0x20	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															mpu_m0 RW 0x0
emac3rx_ecc 0x24	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															
emac3tx_ecc 0x28	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															
nand_ecc 0x2C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															mpu_m0 RW 0x0
nand_read_ecc 0x30	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															mpu_m0 RW 0x0

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
nand_write_ecc 0x34	Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															mpu_m0 RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
onchipram_ecc 0x38	Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															mpu_m0 RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
qspi_ecc 0x3C	Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															mpu_m0 RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
sdmmc_ecc 0x40	Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															mpu_m0 RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
usb0_ecc 0x44	Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															mpu_m0 RW 0x0

Register Address Offset	Bit Fields															
usb1_ecc 0x48	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															mpu_m0 RW 0x0
clock_manager 0x4C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0
fpga_manager_register 0x50	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved							ahb_ap RW 0x0	Reserved							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0
pin_mux_register 0x54	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0

Register Address Offset	Bit Fields															
reset_manager 0x58	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0
system_manager 0x5C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0
osc0_timer 0x60	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0
osc1_timer 0x64	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
watchdog0 0x68	Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
watchdog1 0x6C	Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dap 0x70	Reserved						etr RW 0x0	ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
fpga_manager_streaming 0x74	Reserved							ahb_ap RW 0x0	Reserved							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
security_manager_streaming 0x78	Reserved							ahb_ap RW 0x0	Reserved							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hmc_register 0x7C	Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hmc_adaptor_register 0x80	Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
l3_interconnect_register 0x84	Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															mpu_m0 RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ddr_scheduler_register 0x88	Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															mpu_m0 RW 0x0

Register Address Offset	Bit Fields															
l4_interconnect_firewall_csr 0x8C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved							ahb_ap 0x0	Reserved							fpga2soc 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															mpu_m0 0x0
l4_interconnect_probes_csr 0x90	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															mpu_m0 RW 0x0
l4_qos_csr 0x94	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															mpu_m0 RW 0x0

dma_ecc

Per-Master Security bit for dma_ecc

Module Instance	Base Address	Register Address
noc_fw_l4_sys_l4_sys_scr	0xFFD13100	0xFFD13108

Offset: 0x8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															mpu_m0 RW 0x0

dma_ecc Fields

Bit	Name	Description	Access	Reset
24	ahb_ap	Security bit configuration for transactions from ahb_ap to dma_ecc. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
16	fpga2soc	Security bit configuration for transactions from fpga2soc to dma_ecc. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
0	mpu_m0	Security bit configuration for transactions from mpu_m0 to dma_ecc. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

emac0rx_ecc

Per-Master Security bit for emac0rx_ecc

Module Instance	Base Address	Register Address
noc_fw_l4_sys_l4_sys_scr	0xFFD13100	0xFFD1310C

Offset: 0xC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															mpu_m0 RW 0x0

emac0rx_ecc Fields

Bit	Name	Description	Access	Reset
24	ahb_ap	Security bit configuration for transactions from ahb_ap to emac0rx_ecc. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
16	fpga2soc	Security bit configuration for transactions from fpga2soc to emac0rx_ecc. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
0	mpu_m0	Security bit configuration for transactions from mpu_m0 to emac0rx_ecc. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

emac0tx_ecc

Per-Master Security bit for emac0tx_ecc

Module Instance	Base Address	Register Address
noc_fw_l4_sys_l4_sys_scr	0xFFD13100	0xFFD13110

Offset: 0x10

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															mpu_m0 RW 0x0

emac0tx_ecc Fields

Bit	Name	Description	Access	Reset
24	ahb_ap	Security bit configuration for transactions from ahb_ap to emac0tx_ecc. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
16	fpga2soc	Security bit configuration for transactions from fpga2soc to emac0tx_ecc. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
0	mpu_m0	Security bit configuration for transactions from mpu_m0 to emac0tx_ecc. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

emac1rx_ecc

Per-Master Security bit for emac1rx_ecc

Module Instance	Base Address	Register Address
noc_fw_l4_sys_l4_sys_scr	0xFFD13100	0xFFD13114

Offset: 0x14

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															mpu_m0 RW 0x0

emac1rx_ecc Fields

Bit	Name	Description	Access	Reset
24	ahb_ap	Security bit configuration for transactions from ahb_ap to emac1rx_ecc. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
16	fpga2soc	Security bit configuration for transactions from fpga2soc to emac1rx_ecc. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
0	mpu_m0	Security bit configuration for transactions from mpu_m0 to emac1rx_ecc. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

emac1tx_ecc

Per-Master Security bit for emac1tx_ecc

Module Instance	Base Address	Register Address
noc_fw_l4_sys_l4_sys_scr	0xFFD13100	0xFFD13118

Offset: 0x18

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															mpu_m0 RW 0x0

emac1tx_ecc Fields

Bit	Name	Description	Access	Reset
24	ahb_ap	Security bit configuration for transactions from ahb_ap to emac1tx_ecc. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
16	fpga2soc	Security bit configuration for transactions from fpga2soc to emac1tx_ecc. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

Bit	Name	Description	Access	Reset
0	mpu_m0	Security bit configuration for transactions from mpu_m0 to emac1tx_ecc. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

emac2rx_ecc

Per-Master Security bit for emac2rx_ecc

Module Instance	Base Address	Register Address
noc_fw_l4_sys_l4_sys_scr	0xFFD13100	0xFFD1311C

Offset: 0x1C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															mpu_m0 RW 0x0

emac2rx_ecc Fields

Bit	Name	Description	Access	Reset
24	ahb_ap	Security bit configuration for transactions from ahb_ap to emac2rx_ecc. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

Bit	Name	Description	Access	Reset
16	fpga2soc	Security bit configuration for transactions from fpga2soc to emac2rx_ecc. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
0	mpu_m0	Security bit configuration for transactions from mpu_m0 to emac2rx_ecc. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

emac2tx_ecc

Per-Master Security bit for emac2tx_ecc

Module Instance	Base Address	Register Address
noc_fw_l4_sys_l4_sys_scr	0xFFD13100	0xFFD13120

Offset: 0x20

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															mpu_m0 RW 0x0

emac2tx_ecc Fields

Bit	Name	Description	Access	Reset
24	ahb_ap	Security bit configuration for transactions from ahb_ap to emac2tx_ecc. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
16	fpga2soc	Security bit configuration for transactions from fpga2soc to emac2tx_ecc. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
0	mpu_m0	Security bit configuration for transactions from mpu_m0 to emac2tx_ecc. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

emac3rx_ecc

Per-Master Security bit for emac3rx_ecc

Module Instance	Base Address	Register Address
noc_fw_l4_sys_l4_sys_scr	0xFFD13100	0xFFD13124

Offset: 0x24

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															

emac3rx_ecc Fields

Bit	Name	Description	Access	Reset
31:0	Reserved		RO	0x0

emac3tx_ecc

Per-Master Security bit for emac3tx_ecc

Module Instance	Base Address	Register Address
noc_fw_l4_sys_l4_sys_scr	0xFFD13100	0xFFD13128

Offset: 0x28

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															

emac3tx_ecc Fields

Bit	Name	Description	Access	Reset
31:0	Reserved		RO	0x0

nand_ecc

Per-Master Security bit for nand_ecc

Module Instance	Base Address	Register Address
noc_fw_l4_sys_l4_sys_scr	0xFFD13100	0xFFD1312C

Offset: 0x2C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															mpu_m0 RW 0x0

nand_ecc Fields

Bit	Name	Description	Access	Reset
24	ahb_ap	Security bit configuration for transactions from ahb_ap to nand_ecc. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
16	fpga2soc	Security bit configuration for transactions from fpga2soc to nand_ecc. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
0	mpu_m0	Security bit configuration for transactions from mpu_m0 to nand_ecc. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

nand_read_ecc

Per-Master Security bit for nand_read_ecc

Module Instance	Base Address	Register Address
noc_fw_l4_sys_l4_sys_scr	0xFFD13100	0xFFD13130

Offset: 0x30

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															mpu_m0 RW 0x0

nand_read_ecc Fields

Bit	Name	Description	Access	Reset
24	ahb_ap	Security bit configuration for transactions from ahb_ap to nand_read_ecc. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
16	fpga2soc	Security bit configuration for transactions from fpga2soc to nand_read_ecc. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
0	mpu_m0	Security bit configuration for transactions from mpu_m0 to nand_read_ecc. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

nand_write_ecc

Per-Master Security bit for nand_write_ecc

Module Instance	Base Address	Register Address
noc_fw_l4_sys_l4_sys_scr	0xFFD13100	0xFFD13134

Offset: 0x34

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															mpu_m0 RW 0x0

nand_write_ecc Fields

Bit	Name	Description	Access	Reset
24	ahb_ap	Security bit configuration for transactions from ahb_ap to nand_write_ecc. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
16	fpga2soc	Security bit configuration for transactions from fpga2soc to nand_write_ecc. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

Bit	Name	Description	Access	Reset
0	mpu_m0	Security bit configuration for transactions from mpu_m0 to nand_write_ecc. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

onchipram_ecc

Per-Master Security bit for onchipram_ecc

Module Instance	Base Address	Register Address
noc_fw_l4_sys_l4_sys_scr	0xFFD13100	0xFFD13138

Offset: 0x38

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															mpu_m0 RW 0x0

onchipram_ecc Fields

Bit	Name	Description	Access	Reset
24	ahb_ap	Security bit configuration for transactions from ahb_ap to onchipram_ecc. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

Bit	Name	Description	Access	Reset
16	fpga2soc	Security bit configuration for transactions from fpga2soc to onchipram_ecc. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
0	mpu_m0	Security bit configuration for transactions from mpu_m0 to onchipram_ecc. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

qspi_ecc

Per-Master Security bit for qspi_ecc

Module Instance	Base Address	Register Address
noc_fw_l4_sys_l4_sys_scr	0xFFD13100	0xFFD1313C

Offset: 0x3C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															mpu_m0 RW 0x0

qspi_ecc Fields

Bit	Name	Description	Access	Reset
24	ahb_ap	Security bit configuration for transactions from ahb_ap to qspi_ecc. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
16	fpga2soc	Security bit configuration for transactions from fpga2soc to qspi_ecc. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
0	mpu_m0	Security bit configuration for transactions from mpu_m0 to qspi_ecc. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

sdmmc_ecc

Per-Master Security bit for sdmmc_ecc

Module Instance	Base Address	Register Address
noc_fw_l4_sys_l4_sys_scr	0xFFD13100	0xFFD13140

Offset: 0x40

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															mpu_m0 RW 0x0

sdmmc_ecc Fields

Bit	Name	Description	Access	Reset
24	ahb_ap	Security bit configuration for transactions from ahb_ap to sdmmc_ecc. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
16	fpga2soc	Security bit configuration for transactions from fpga2soc to sdmmc_ecc. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
0	mpu_m0	Security bit configuration for transactions from mpu_m0 to sdmmc_ecc. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

usb0_ecc

Per-Master Security bit for usb0_ecc

Module Instance	Base Address	Register Address
noc_fw_l4_sys_l4_sys_scr	0xFFD13100	0xFFD13144

Offset: 0x44

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															mpu_m0 RW 0x0

usb0_ecc Fields

Bit	Name	Description	Access	Reset
24	ahb_ap	Security bit configuration for transactions from ahb_ap to usb0_ecc. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
16	fpga2soc	Security bit configuration for transactions from fpga2soc to usb0_ecc. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
0	mpu_m0	Security bit configuration for transactions from mpu_m0 to usb0_ecc. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

usb1_ecc

Per-Master Security bit for usb1_ecc

Module Instance	Base Address	Register Address
noc_fw_l4_sys_l4_sys_scr	0xFFD13100	0xFFD13148

Offset: 0x48

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															mpu_m0 RW 0x0

usb1_ecc Fields

Bit	Name	Description	Access	Reset
24	ahb_ap	Security bit configuration for transactions from ahb_ap to usb1_ecc. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
16	fpga2soc	Security bit configuration for transactions from fpga2soc to usb1_ecc. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
0	mpu_m0	Security bit configuration for transactions from mpu_m0 to usb1_ecc. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

clock_manager

Per-Master Security bit for clock_manager

Module Instance	Base Address	Register Address
noc_fw_l4_sys_l4_sys_scr	0xFFD13100	0xFFD1314C

Offset: 0x4C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0

clock_manager Fields

Bit	Name	Description	Access	Reset
24	ahb_ap	Security bit configuration for transactions from ahb_ap to clock_manager. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
16	fpga2soc	Security bit configuration for transactions from fpga2soc to clock_manager. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
8	dma	Security bit configuration for transactions from dma to clock_manager. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

Bit	Name	Description	Access	Reset
0	mpu_m0	Security bit configuration for transactions from mpu_m0 to clock_manager. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

fpga_manager_register

Per-Master Security bit for fpga_manager

Module Instance	Base Address	Register Address
noc_fw_l4_sys_l4_sys_scr	0xFFD13100	0xFFD13150

Offset: 0x50

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							ahb_ap RW 0x0	Reserved							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0

fpga_manager_register Fields

Bit	Name	Description	Access	Reset
24	ahb_ap	Security bit configuration for transactions from ahb_ap to fpga_manager. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

Bit	Name	Description	Access	Reset
8	dma	Security bit configuration for transactions from dma to fpga_manager. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
0	mpu_m0	Security bit configuration for transactions from mpu_m0 to fpga_manager. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

pin_mux_register

Per-Master Security bit for pin_mux_register

Module Instance	Base Address	Register Address
noc_fw_l4_sys_l4_sys_scr	0xFFD13100	0xFFD13154

Offset: 0x54

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0

pin_mux_register Fields

Bit	Name	Description	Access	Reset
24	ahb_ap	Security bit configuration for transactions from ahb_ap to pin_mux_register. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
16	fpga2soc	Security bit configuration for transactions from fpga2soc to pin_mux_register. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
8	dma	Security bit configuration for transactions from dma to pin_mux_register. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
0	mpu_m0	Security bit configuration for transactions from mpu_m0 to pin_mux_register. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

reset_manager

Per-Master Security bit for reset_manager

Module Instance	Base Address	Register Address
noc_fw_l4_sys_l4_sys_scr	0xFFD13100	0xFFD13158

Offset: 0x58

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0

reset_manager Fields

Bit	Name	Description	Access	Reset
24	ahb_ap	Security bit configuration for transactions from ahb_ap to reset_manager. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
16	fpga2soc	Security bit configuration for transactions from fpga2soc to reset_manager. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
8	dma	Security bit configuration for transactions from dma to reset_manager. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
0	mpu_m0	Security bit configuration for transactions from mpu_m0 to reset_manager. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

system_manager

Per-Master Security bit for system_manager

Module Instance	Base Address	Register Address
noc_fw_l4_sys_l4_sys_scr	0xFFD13100	0xFFD1315C

Offset: 0x5C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0

system_manager Fields

Bit	Name	Description	Access	Reset
24	ahb_ap	Security bit configuration for transactions from ahb_ap to system_manager. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
16	fpga2soc	Security bit configuration for transactions from fpga2soc to system_manager. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
8	dma	Security bit configuration for transactions from dma to system_manager. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

Bit	Name	Description	Access	Reset
0	mpu_m0	Security bit configuration for transactions from mpu_m0 to system_manager. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

osc0_timer

Per-Master Security bit for osc0_timer

Module Instance	Base Address	Register Address
noc_fw_l4_sys_l4_sys_scr	0xFFD13100	0xFFD13160

Offset: 0x60

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0

osc0_timer Fields

Bit	Name	Description	Access	Reset
24	ahb_ap	Security bit configuration for transactions from ahb_ap to osc0_timer. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

Bit	Name	Description	Access	Reset
16	fpga2soc	Security bit configuration for transactions from fpga2soc to osc0_timer. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
8	dma	Security bit configuration for transactions from dma to osc0_timer. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
0	mpu_m0	Security bit configuration for transactions from mpu_m0 to osc0_timer. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

osc1_timer

Per-Master Security bit for osc1_timer

Module Instance	Base Address	Register Address
noc_fw_l4_sys_l4_sys_scr	0xFFD13100	0xFFD13164

Offset: 0x64

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0

osc1_timer Fields

Bit	Name	Description	Access	Reset
24	ahb_ap	Security bit configuration for transactions from ahb_ap to osc1_timer. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
16	fpga2soc	Security bit configuration for transactions from fpga2soc to osc1_timer. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
8	dma	Security bit configuration for transactions from dma to osc1_timer. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
0	mpu_m0	Security bit configuration for transactions from mpu_m0 to osc1_timer. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

watchdog0

Per-Master Security bit for watchdog0

Module Instance	Base Address	Register Address
noc_fw_l4_sys_l4_sys_scr	0xFFD13100	0xFFD13168

Offset: 0x68

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0

watchdog0 Fields

Bit	Name	Description	Access	Reset
24	ahb_ap	Security bit configuration for transactions from ahb_ap to watchdog0. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
16	fpga2soc	Security bit configuration for transactions from fpga2soc to watchdog0. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
8	dma	Security bit configuration for transactions from dma to watchdog0. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

Bit	Name	Description	Access	Reset
0	mpu_m0	Security bit configuration for transactions from mpu_m0 to watchdog0. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

watchdog1

Per-Master Security bit for watchdog1

Module Instance	Base Address	Register Address
noc_fw_l4_sys_l4_sys_scr	0xFFD13100	0xFFD1316C

Offset: 0x6C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0

watchdog1 Fields

Bit	Name	Description	Access	Reset
24	ahb_ap	Security bit configuration for transactions from ahb_ap to watchdog1. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

Bit	Name	Description	Access	Reset
16	fpga2soc	Security bit configuration for transactions from fpga2soc to watchdog1. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
8	dma	Security bit configuration for transactions from dma to watchdog1. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
0	mpu_m0	Security bit configuration for transactions from mpu_m0 to watchdog1. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

dap

Per-Master Security bit for dap

Module Instance	Base Address	Register Address
noc_fw_l4_sys_l4_sys_scr	0xFFD13100	0xFFD13170

Offset: 0x70

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved						etr RW 0x0	ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0

dap Fields

Bit	Name	Description	Access	Reset
25	etr	Security bit configuration for transactions from etr to dap. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
24	ahb_ap	Security bit configuration for transactions from ahb_ap to dap. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
16	fpga2soc	Security bit configuration for transactions from fpga2soc to dap. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
8	dma	Security bit configuration for transactions from dma to dap. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
0	mpu_m0	Security bit configuration for transactions from mpu_m0 to dap. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

fpga_manager_streaming

Per-Master Security bit for fpga_manager_streaming

Module Instance	Base Address	Register Address
noc_fw_l4_sys_l4_sys_scr	0xFFD13100	0xFFD13174

Offset: 0x74

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							ahb_ap RW 0x0	Reserved							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0

fpga_manager_streaming Fields

Bit	Name	Description	Access	Reset
24	ahb_ap	Security bit configuration for transactions from ahb_ap to fpga_manager_streaming. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
8	dma	Security bit configuration for transactions from dma to fpga_manager_streaming. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

Bit	Name	Description	Access	Reset
0	mpu_m0	Security bit configuration for transactions from mpu_m0 to fpga_manager_streaming. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

security_manager_streaming

Per-Master Security bit for security_manager_streaming

Module Instance	Base Address	Register Address
noc_fw_l4_sys_l4_sys_scr	0xFFD13100	0xFFD13178

Offset: 0x78

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							ahb_ap RW 0x0	Reserved							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0

security_manager_streaming Fields

Bit	Name	Description	Access	Reset
24	ahb_ap	Security bit configuration for transactions from ahb_ap to security_manager_streaming. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
8	dma	Security bit configuration for transactions from dma to security_manager_streaming. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
0	mpu_m0	Security bit configuration for transactions from mpu_m0 to security_manager_streaming. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

hmc_register

Per-Master Security bit for hmc_register

Module Instance	Base Address	Register Address
noc_fw_l4_sys_l4_sys_scr	0xFFD13100	0xFFD1317C

Offset: 0x7C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0

hmc_register Fields

Bit	Name	Description	Access	Reset
24	ahb_ap	Security bit configuration for transactions from ahb_ap to hmc_register. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
16	fpga2soc	Security bit configuration for transactions from fpga2soc to hmc_register. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
8	dma	Security bit configuration for transactions from dma to hmc_register. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
0	mpu_m0	Security bit configuration for transactions from mpu_m0 to hmc_register. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

hmc_adaptor_register

Per-Master Security bit for hmc_adaptor_register

Module Instance	Base Address	Register Address
noc_fw_l4_sys_l4_sys_scr	0xFFD13100	0xFFD13180

Offset: 0x80

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0

hmc_adaptor_register Fields

Bit	Name	Description	Access	Reset
24	ahb_ap	Security bit configuration for transactions from ahb_ap to hmc_adaptor_register. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
16	fpga2soc	Security bit configuration for transactions from fpga2soc to hmc_adaptor_register. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
8	dma	Security bit configuration for transactions from dma to hmc_adaptor_register. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

Bit	Name	Description	Access	Reset
0	mpu_m0	Security bit configuration for transactions from mpu_m0 to hmc_adaptor_register. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

l3_interconnect_register

Per-Master Security bit for ddr_scheduler_register

Module Instance	Base Address	Register Address
noc_fw_l4_sys_l4_sys_scr	0xFFD13100	0xFFD13184

Offset: 0x84

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															mpu_m0 RW 0x0

l3_interconnect_register Fields

Bit	Name	Description	Access	Reset
24	ahb_ap	Security bit configuration for transactions from ahb_ap to ddr_schedule_register. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
16	fpga2soc	Security bit configuration for transactions from fpga2soc to ddr_scheduler_register. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
0	mpu_m0	Security bit configuration for transactions from mpu_m0 to ddr_scheduler_register. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

ddr_scheduler_register

Per-Master Security bit for ddr_scheduler_register

Module Instance	Base Address	Register Address
noc_fw_l4_sys_l4_sys_scr	0xFFD13100	0xFFD13188

Offset: 0x88

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															mpu_m0 RW 0x0

ddr_scheduler_register Fields

Bit	Name	Description	Access	Reset
24	ahb_ap	Security bit configuration for transactions from ahb_ap to ddr_schedule_register. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
16	fpga2soc	Security bit configuration for transactions from fpga2soc to ddr_scheduler_register. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
0	mpu_m0	Security bit configuration for transactions from mpu_m0 to ddr_scheduler_register. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

l4_interconnect_firewall_csr

Per-Master Security bit for noc_firewall_scr

Module Instance	Base Address	Register Address
noc_fw_l4_sys_l4_sys_scr	0xFFD13100	0xFFD1318C

Offset: 0x8C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							ahb_ap 0x0	Reserved							fpga2soc 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															mpu_m0 0x0

l4_interconnect_firewall_csr Fields

Bit	Name	Description	Access	Reset
24	ahb_ap	Security bit configuration for transactions from ahb_ap to noc_firewall_scr. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
16	fpga2soc	Security bit configuration for transactions from fpga2soc to noc_firewall_scr. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
0	mpu_m0	Security bit configuration for transactions from mpu_m0 to noc_firewall_scr. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

l4_interconnect_probes_csr

Per-Master Security bit for noc_probes_register

Module Instance	Base Address	Register Address
noc_fw_l4_sys_l4_sys_scr	0xFFD13100	0xFFD13190

Offset: 0x90

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															mpu_m0 RW 0x0

l4_interconnect_probes_csr Fields

Bit	Name	Description	Access	Reset
24	ahb_ap	Security bit configuration for transactions from ahb_ap to noc_probes_register. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
16	fpga2soc	Security bit configuration for transactions from fpga2soc to noc_probes_register. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
0	mpu_m0	Security bit configuration for transactions from mpu_m0 to noc_probes_register. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

l4_qos_csr

Per-Master Security bit for noc_probes_register

Module Instance	Base Address	Register Address
noc_fw_l4_sys_l4_sys_scr	0xFFD13100	0xFFD13194

Offset: 0x94

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							ahb_ap RW 0x0	Reserved							fpga2soc RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															mpu_m0 RW 0x0

l4_qos_csr Fields

Bit	Name	Description	Access	Reset
24	ahb_ap	Security bit configuration for transactions from ahb_ap to noc_probes_register. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
16	fpga2soc	Security bit configuration for transactions from fpga2soc to noc_probes_register. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

Bit	Name	Description	Access	Reset
0	mpu_m0	Security bit configuration for transactions from mpu_m0 to noc_probes_register. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

noc_fw_ocram_ocram_scr Address Map

Module Instance	Base Address	End Address
noc_fw_ocram_ocram_scr	0xFFD13200	0xFFD132FF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
enable on page 7-414	0x0	32	RW	0x0	Enable
enable_set on page 7-415	0x4	32	WO	0x0	Sets Region Enable field when written with 1
enable_clear on page 7-416	0x8	32	WO	0x0	Clears Region Enable field when written with 1
region0addr on page 7-417	0xC	32	RW	0x0	Base and Limit definition for Region 0
region1addr on page 7-418	0x10	32	RW	0x0	Base and Limit definition for Region 1
region2addr on page 7-419	0x14	32	RW	0x0	Base and Limit definition for Region 2
region3addr on page 7-420	0x18	32	RW	0x0	Base and Limit definition for Region 3
region4addr on page 7-421	0x1C	32	RW	0x0	Base and Limit definition for Region 4

Register	Offset	Width	Access	Reset Value	Description
region5addr on page 7-422	0x20	32	RW	0x0	Base and Limit definition for Region 5

noc_fw_ocram_ocram_scr Summary

Base Address: 0xFFD13200

Register Address Offset	Bit Fields															
noc_fw_ocram_ocram_scr																
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
enable 0x0	Reserved										region5enable RW 0x0	region4enable RW 0x0	region3enable RW 0x0	region2enable RW 0x0	region1enable RW 0x0	region0enable RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
enable_set 0x4	Reserved										region5enable WO 0x0	region4enable WO 0x0	region3enable WO 0x0	region2enable WO 0x0	region1enable WO 0x0	region0enable WO 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
enable_clear 0x8	Reserved										region5enable WO 0x0	region4enable WO 0x0	region3enable WO 0x0	region2enable WO 0x0	region1enable WO 0x0	region0enable WO 0x0

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
region0addr 0xC	Reserved										limit RW 0x0					
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved										base RW 0x0					
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
region1addr 0x10	Reserved										limit RW 0x0					
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved										base RW 0x0					
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
region2addr 0x14	Reserved										limit RW 0x0					
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved										base RW 0x0					
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
region3addr 0x18	Reserved										limit RW 0x0					
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved										base RW 0x0					
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
region4addr 0x1C	Reserved										limit RW 0x0					
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved										base RW 0x0					
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
region5addr 0x20	Reserved										limit RW 0x0					
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved										base RW 0x0					

enable

Enable

Module Instance	Base Address	Register Address
noc_fw_ocram_ocram_scr	0xFFD13200	0xFFD13200

Offset: 0x0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										region5enable RW 0x0	region4enable RW 0x0	region3enable RW 0x0	region2enable RW 0x0	region1enable RW 0x0	region0enable RW 0x0

enable Fields

Bit	Name	Description	Access	Reset
5	region5enable	Region 5 Enable. Value of 1 means region is enabled, Value of 0 means region is disabled	RW	0x0
4	region4enable	Region 4 Enable. Value of 1 means region is enabled, Value of 0 means region is disabled	RW	0x0
3	region3enable	Region 3 Enable. Value of 1 means region is enabled, Value of 0 means region is disabled	RW	0x0
2	region2enable	Region 2 Enable. Value of 1 means region is enabled, Value of 0 means region is disabled	RW	0x0

Bit	Name	Description	Access	Reset
1	region1enable	Region 1 Enable. Value of 1 means region is enabled, Value of 0 means region is disabled	RW	0x0
0	region0enable	Region 0 Enable. Value of 1 means region is enabled, Value of 0 means region is disabled	RW	0x0

enable_set

Sets Region Enable field when written with 1

Module Instance	Base Address	Register Address
noc_fw_ocram_ocram_scr	0xFFD13200	0xFFD13204

Offset: 0x4

Access: WO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										region5enable WO 0x0	region4enable WO 0x0	region3enable WO 0x0	region2enable WO 0x0	region1enable WO 0x0	region0enable WO 0x0

enable_set Fields

Bit	Name	Description	Access	Reset
5	region5enable	Region 5 Enable Set Writing zero has no effect Writing one will set the region5enable bit to one	WO	0x0

Bit	Name	Description	Access	Reset
4	region4enable	Region 4 Enable Set Writing zero has no effect Writing one will set the region4enable bit to one	WO	0x0
3	region3enable	Region 3 Enable Set Writing zero has no effect Writing one will set the region3enable bit to one	WO	0x0
2	region2enable	Region 2 Enable Set Writing zero has no effect Writing one will set the region2enable bit to one	WO	0x0
1	region1enable	Region 1 Enable Set Writing zero has no effect Writing one will set the region1enable bit to one	WO	0x0
0	region0enable	Region 0 Enable Set Writing zero has no effect Writing one will set the region0enable bit to one	WO	0x0

enable_clear

Clears Region Enable field when written with 1

Module Instance	Base Address	Register Address
noc_fw_ocram_ocram_scr	0xFFD13200	0xFFD13208

Offset: 0x8

Access: WO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										region5enable WO 0x0	region4enable WO 0x0	region3enable WO 0x0	region2enable WO 0x0	region1enable WO 0x0	region0enable WO 0x0

enable_clear Fields

Bit	Name	Description	Access	Reset
5	region5enable	Region 5 Enable Clear Writing zero has no effect Writing one will clear the region5enable bit to zero	WO	0x0
4	region4enable	Region 4 Enable Clear Writing zero has no effect Writing one will clear the region4enable bit to zero	WO	0x0
3	region3enable	Region 3 Enable Clear Writing zero has no effect Writing one will clear the region3enable bit to zero	WO	0x0
2	region2enable	Region 2 Enable Clear Writing zero has no effect Writing one will clear the region2enable bit to zero	WO	0x0
1	region1enable	Region 1 Enable Clear Writing zero has no effect Writing one will clear the region1enable bit to zero	WO	0x0
0	region0enable	Region 0 Enable Clear Writing zero has no effect Writing one will clear the region0enable bit to zero	WO	0x0

region0addr

Base and Limit definition for Region 0

Module Instance	Base Address	Register Address
noc_fw_ocram_ocram_scr	0xFFD13200	0xFFD1320C

Offset: 0xC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved										limit RW 0x0					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										base RW 0x0					

region0addr Fields

Bit	Name	Description	Access	Reset
21:16	limit	Limit defines the 6 bit MSB of the address field. Remaining LSB field is all ones. Region end address is {limit, 12'hFFF}	RW	0x0
5:0	base	Base defines the 6 bit MSB of the address field. Remaining LSB field is all zeros. Region start address is {base, 12'h000}	RW	0x0

region1addr

Base and Limit definition for Region 1

Module Instance	Base Address	Register Address
noc_fw_ocram_ocram_scr	0xFFD13200	0xFFD13210

Offset: 0x10

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved										limit RW 0x0					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										base RW 0x0					

region1addr Fields

Bit	Name	Description	Access	Reset
21:16	limit	Limit defines the 6 bit MSB of the address field. Remaining LSB field is all ones. Region end address is {limit, 12'hFFF}	RW	0x0
5:0	base	Base defines the 6 bit MSB of the address field. Remaining LSB field is all zeros. Region start address is {base, 12'h000}	RW	0x0

region2addr

Base and Limit definition for Region 2

Module Instance	Base Address	Register Address
noc_fw_ocram_ocram_scr	0xFFD13200	0xFFD13214

Offset: 0x14

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved										limit RW 0x0					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										base RW 0x0					

region2addr Fields

Bit	Name	Description	Access	Reset
21:16	limit	Limit defines the 6 bit MSB of the address field. Remaining LSB field is all ones. Region end address is {limit, 12'hFFF}	RW	0x0
5:0	base	Base defines the 6 bit MSB of the address field. Remaining LSB field is all zeros. Region start address is {base, 12'h000}	RW	0x0

region3addr

Base and Limit definition for Region 3

Module Instance	Base Address	Register Address
noc_fw_ocram_ocram_scr	0xFFD13200	0xFFD13218

Offset: 0x18

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved										limit RW 0x0					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										base RW 0x0					

region3addr Fields

Bit	Name	Description	Access	Reset
21:16	limit	Limit defines the 6 bit MSB of the address field. Remaining LSB field is all ones. Region end address is {limit, 12'hFFF}	RW	0x0
5:0	base	Base defines the 6 bit MSB of the address field. Remaining LSB field is all zeros. Region start address is {base, 12'h000}	RW	0x0

region4addr

Base and Limit definition for Region 4

Module Instance	Base Address	Register Address
noc_fw_ocram_ocram_scr	0xFFD13200	0xFFD1321C

Offset: 0x1C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved										limit RW 0x0					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										base RW 0x0					

region4addr Fields

Bit	Name	Description	Access	Reset
21:16	limit	Limit defines the 6 bit MSB of the address field. Remaining LSB field is all ones. Region end address is {limit, 12'hFFF}	RW	0x0
5:0	base	Base defines the 6 bit MSB of the address field. Remaining LSB field is all zeros. Region start address is {base, 12'h000}	RW	0x0

region5addr

Base and Limit definition for Region 5

Module Instance	Base Address	Register Address
noc_fw_ocram_ocram_scr	0xFFD13200	0xFFD13220

Offset: 0x20

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved										limit RW 0x0					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										base RW 0x0					

region5addr Fields

Bit	Name	Description	Access	Reset
21:16	limit	Limit defines the 6 bit MSB of the address field. Remaining LSB field is all ones. Region end address is {limit, 12'hFFF}	RW	0x0
5:0	base	Base defines the 6 bit MSB of the address field. Remaining LSB field is all zeros. Region start address is {base, 12'h000}	RW	0x0

noc_fw_ddr_mpu_fpga2sdram_ddr_scr Address Map

Module Instance	Base Address	End Address
noc_fw_ddr_mpu_fpga2sdram_ddr_scr	0xFFD13300	0xFFD133FF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
enable on page 7-429	0x0	32	RW	0x0	Enable
enable_set on page 7-431	0x4	32	WO	0x0	Sets Master Region Enable field when written with 1

Register	Offset	Width	Access	Reset Value	Description
enable_clear on page 7-433	0x8	32	WO	0x0	Clears Master Region Enable field when written with 1
mpuregion0addr on page 7-436	0x10	32	RW	0x0	Base and Limit definition for MPU Region 0
mpuregion1addr on page 7-436	0x14	32	RW	0x0	Base and Limit definition for MPU Region 1
mpuregion2addr on page 7-437	0x18	32	RW	0x0	Base and Limit definition for MPU Region 2
mpuregion3addr on page 7-438	0x1C	32	RW	0x0	Base and Limit definition for MPU Region 3
fpga2sdram0region0addr on page 7-439	0x20	32	RW	0x0	Base and Limit definition for FPGA2SDRAM0 Region 0
fpga2sdram0region1addr on page 7-440	0x24	32	RW	0x0	Base and Limit definition for FPGA2SDRAM0 Region 1
fpga2sdram0region2addr on page 7-441	0x28	32	RW	0x0	Base and Limit definition for FPGA2SDRAM0 Region 2
fpga2sdram0region3addr on page 7-441	0x2C	32	RW	0x0	Base and Limit definition for FPGA2SDRAM0 Region 3
fpga2sdram1region0addr on page 7-442	0x30	32	RW	0x0	Base and Limit definition for FPGA2SDRAM1 Region 0
fpga2sdram1region1addr on page 7-443	0x34	32	RW	0x0	Base and Limit definition for FPGA2SDRAM1 Region 1
fpga2sdram1region2addr on page 7-444	0x38	32	RW	0x0	Base and Limit definition for FPGA2SDRAM1 Region 2

Register	Offset	Width	Access	Reset Value	Description
fpga2sdram1region3addr on page 7-445	0x3C	32	RW	0x0	Base and Limit definition for FPGA2SDRAM1 Region 3
fpga2sdram2region0addr on page 7-446	0x40	32	RW	0x0	Base and Limit definition for FPGA2SDRAM2 Region 0
fpga2sdram2region1addr on page 7-446	0x44	32	RW	0x0	Base and Limit definition for FPGA2SDRAM2 Region 1
fpga2sdram2region2addr on page 7-447	0x48	32	RW	0x0	Base and Limit definition for FPGA2SDRAM2 Region 2
fpga2sdram2region3addr on page 7-448	0x4C	32	RW	0x0	Base and Limit definition for FPGA2SDRAM2 Region 3

noc_fw_ddr_mpu_fpga2sdram_ddr_scr Summary

Base Address: 0xFFD13300

Register Address Offset	Bit Fields															
noc_fw_ddr_mpu_fpga2sdram_ddr_scr																
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
enable 0x0	fpga2sdram2regionen3enable RW 0x0	fpga2sdram2regionen2enable RW 0x0	fpga2sdram2regionen1enable RW 0x0	fpga2sdram2regionen0enable RW 0x0	fpga2sdram1regionen3enable RW 0x0	fpga2sdram1regionen2enable RW 0x0	fpga2sdram1regionen1enable RW 0x0	fpga2sdram1regionen0enable RW 0x0	fpga2sdram0regionen3enable RW 0x0	fpga2sdram0regionen2enable RW 0x0	fpga2sdram0regionen1enable RW 0x0	fpga2sdram0regionen0enable RW 0x0	mpuregionen3enable RW 0x0	mpuregionen2enable RW 0x0	mpuregionen1enable RW 0x0	mpuregionenable RW 0x0

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
enable_set 0x4	fpga2sdram2regionenable WO 0x0	fpga2sdram2regionenable WO 0x0	fpga2sdram2regionenable WO 0x0	fpga2sdram2regionenable WO 0x0	fpga2sdram1regionenable WO 0x0	fpga2sdram1regionenable WO 0x0	fpga2sdram1regionenable WO 0x0	fpga2sdram1regionenable WO 0x0	fpga2sdram0regionenable WO 0x0	fpga2sdram0regionenable WO 0x0	fpga2sdram0regionenable WO 0x0	fpga2sdram0regionenable WO 0x0	mpuregion3enable WO 0x0	mpuregion2enable WO 0x0	mpuregion1enable WO 0x0	mpuregion0enable WO 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
enable_clear 0x8	fpga2sdram2regionenable WO 0x0	fpga2sdram2regionenable WO 0x0	fpga2sdram2regionenable WO 0x0	fpga2sdram2regionenable WO 0x0	fpga2sdram1regionenable WO 0x0	fpga2sdram1regionenable WO 0x0	fpga2sdram1regionenable WO 0x0	fpga2sdram1regionenable WO 0x0	fpga2sdram0regionenable WO 0x0	fpga2sdram0regionenable WO 0x0	fpga2sdram0regionenable WO 0x0	fpga2sdram0regionenable WO 0x0	mpuregion3enable WO 0x0	mpuregion2enable WO 0x0	mpuregion1enable WO 0x0	mpuregion0enable WO 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
mpuregion0addr 0x10	limit RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	base RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
mpuregion1addr 0x14	limit RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	base RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
mpuregion2addr 0x18	limit RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	base RW 0x0															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
mpuregion3addr 0x1C	limit RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	base RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
fpga2sdram0region0addr 0x20	limit RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	base RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
fpga2sdram0region1addr 0x24	limit RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	base RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
fpga2sdram0region2addr 0x28	limit RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	base RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
fpga2sdram0region3addr 0x2C	limit RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	base RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
fpga2sdram1region0addr 0x30	limit RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	base RW 0x0															

Register Address Offset	Bit Fields															
fpga2sdram1region1addr 0x34	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	limit RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	base RW 0x0															
fpga2sdram1region2addr 0x38	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	limit RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	base RW 0x0															
fpga2sdram1region3addr 0x3C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	limit RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	base RW 0x0															
fpga2sdram2region0addr 0x40	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	limit RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	base RW 0x0															
fpga2sdram2region1addr 0x44	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	limit RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	base RW 0x0															
fpga2sdram2region2addr 0x48	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	limit RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	base RW 0x0															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
fpga2sdram2region3addr 0x4C	limit RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	base RW 0x0															

enable

Enable

Module Instance	Base Address	Register Address
noc_fw_ddr_mpu_fpga2sdram_ddr_scr	0xFFD13300	0xFFD13300

Offset: 0x0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
fpga2sdram2region3enable RW 0x0	fpga2sdram2region2enable RW 0x0	fpga2sdram2region1enable RW 0x0	fpga2sdram2region0enable RW 0x0	fpga2sdram1region3enable RW 0x0	fpga2sdram1region2enable RW 0x0	fpga2sdram1region1enable RW 0x0	fpga2sdram1region0enable RW 0x0	fpga2sdram0region3enable RW 0x0	fpga2sdram0region2enable RW 0x0	fpga2sdram0region1enable RW 0x0	fpga2sdram0region0enable RW 0x0	mpuregion3enable RW 0x0	mpuregion2enable RW 0x0	mpuregion1enable RW 0x0	mpuregion0enable RW 0x0

enable Fields

Bit	Name	Description	Access	Reset
15	fpga2sdram2region3enable	FPGA2SDRAM2 Region Enable. Value of 1 means region is enabled, Value of 0 means region is disabled	RW	0x0
14	fpga2sdram2region2enable	FPGA2SDRAM2 Region Enable. Value of 1 means region is enabled, Value of 0 means region is disabled	RW	0x0
13	fpga2sdram2region1enable	FPGA2SDRAM2 Region Enable. Value of 1 means region is enabled, Value of 0 means region is disabled	RW	0x0
12	fpga2sdram2region0enable	FPGA2SDRAM2 Region Enable. Value of 1 means region is enabled, Value of 0 means region is disabled	RW	0x0
11	fpga2sdram1region3enable	FPGA2SDRAM1 Region Enable. Value of 1 means region is enabled, Value of 0 means region is disabled	RW	0x0
10	fpga2sdram1region2enable	FPGA2SDRAM1 Region Enable. Value of 1 means region is enabled, Value of 0 means region is disabled	RW	0x0
9	fpga2sdram1region1enable	FPGA2SDRAM1 Region Enable. Value of 1 means region is enabled, Value of 0 means region is disabled	RW	0x0
8	fpga2sdram1region0enable	FPGA2SDRAM1 Region Enable. Value of 1 means region is enabled, Value of 0 means region is disabled	RW	0x0
7	fpga2sdram0region3enable	FPGA2SDRAM0 Region Enable. Value of 1 means region is enabled, Value of 0 means region is disabled	RW	0x0
6	fpga2sdram0region2enable	FPGA2SDRAM0 Region Enable. Value of 1 means region is enabled, Value of 0 means region is disabled	RW	0x0
5	fpga2sdram0region1enable	FPGA2SDRAM0 Region Enable. Value of 1 means region is enabled, Value of 0 means region is disabled	RW	0x0
4	fpga2sdram0region0enable	FPGA2SDRAM0 Region Enable. Value of 1 means region is enabled, Value of 0 means region is disabled	RW	0x0

Bit	Name	Description	Access	Reset
3	mpuregion3enable	MPU Region 3 Enable. Value of 1 means region is enabled, Value of 0 means region is disabled	RW	0x0
2	mpuregion2enable	MPU Region 2 Enable. Value of 1 means region is enabled, Value of 0 means region is disabled	RW	0x0
1	mpuregion1enable	MPU Region 1 Enable. Value of 1 means region is enabled, Value of 0 means region is disabled	RW	0x0
0	mpuregion0enable	MPU Region 0 Enable. Value of 1 means region is enabled, Value of 0 means region is disabled	RW	0x0

enable_set

Sets Master Region Enable field when written with 1

Module Instance	Base Address	Register Address
noc_fw_ddr_mpu_fpga2sdram_ddr_scr	0xFFD13300	0xFFD13304

Offset: 0x4

Access: WO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
fpga2sdram2region3enable WO 0x0	fpga2sdram2region2enable WO 0x0	fpga2sdram2region1enable WO 0x0	fpga2sdram2region0enable WO 0x0	fpga2sdram1region3enable WO 0x0	fpga2sdram1region2enable WO 0x0	fpga2sdram1region1enable WO 0x0	fpga2sdram1region0enable WO 0x0	fpga2sdram0region3enable WO 0x0	fpga2sdram0region2enable WO 0x0	fpga2sdram0region1enable WO 0x0	fpga2sdram0region0enable WO 0x0	mpuregion3enable WO 0x0	mpuregion2enable WO 0x0	mpuregion1enable WO 0x0	mpuregion0enable WO 0x0

enable_set Fields

Bit	Name	Description	Access	Reset
15	fpga2sdram2region3enable	FPGA2SDRAM2 Region 3 Enable Set. Writing zero has no effect Writing one will set the fpga2sdram2region3enable bit to one	WO	0x0
14	fpga2sdram2region2enable	FPGA2SDRAM2 Region 2 Enable Set. Writing zero has no effect Writing one will set the fpga2sdram2region2enable bit to one	WO	0x0
13	fpga2sdram2region1enable	FPGA2SDRAM2 Region 1 Enable Set. Writing zero has no effect Writing one will set the fpga2sdram2region1enable bit to one	WO	0x0
12	fpga2sdram2region0enable	FPGA2SDRAM2 Region 0 Enable Set. Writing zero has no effect Writing one will set the fpga2sdram2region0enable bit to one	WO	0x0
11	fpga2sdram1region3enable	FPGA2SDRAM1 Region 3 Enable Set. Writing zero has no effect Writing one will set the fpga2sdram1region3enable bit to one	WO	0x0
10	fpga2sdram1region2enable	FPGA2SDRAM1 Region 2 Enable Set. Writing zero has no effect Writing one will set the fpga2sdram1region2enable bit to one	WO	0x0
9	fpga2sdram1region1enable	FPGA2SDRAM1 Region 1 Enable Set. Writing zero has no effect Writing one will set the fpga2sdram1region1enable bit to one	WO	0x0
8	fpga2sdram1region0enable	FPGA2SDRAM1 Region 0 Enable Set. Writing zero has no effect Writing one will set the fpga2sdram1region0enable bit to one	WO	0x0
7	fpga2sdram0region3enable	FPGA2SDRAM0 Region 3 Enable Set. Writing zero has no effect Writing one will set the fpga2sdram0region3enable bit to one	WO	0x0

Bit	Name	Description	Access	Reset
6	fpga2sdram0region2enable	FPGA2SDRAM0 Region 2 Enable Set. Writing zero has no effect Writing one will set the fpga2sdram0region2enable bit to one	WO	0x0
5	fpga2sdram0region1enable	FPGA2SDRAM0 Region 1 Enable Set. Writing zero has no effect Writing one will set the fpga2sdram0region1enable bit to one	WO	0x0
4	fpga2sdram0region0enable	FPGA2SDRAM0 Region 0 Enable Set. Writing zero has no effect Writing one will set the fpga2sdram0region0enable bit to one	WO	0x0
3	mpuregion3enable	MPU Region 3 Enable Set. Writing zero has no effect Writing one will set the mpuregion3enable bit to one	WO	0x0
2	mpuregion2enable	MPU Region 2 Enable Set. Writing zero has no effect Writing one will set the mpuregion2enable bit to one	WO	0x0
1	mpuregion1enable	MPU Region 1 Enable Set. Writing zero has no effect Writing one will set the mpuregion1enable bit to one	WO	0x0
0	mpuregion0enable	MPU Region 0 Enable Set. Writing zero has no effect Writing one will set the mpuregion0enable bit to one	WO	0x0

enable_clear

Clears Master Region Enable field when written with 1

Module Instance	Base Address	Register Address
noc_fw_ddr_mpu_fpga2sdram_ddr_scr	0xFFD13300	0xFFD13308

Offset: 0x8

Access: WO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
fpga2sdram2region3enable WO 0x0	fpga2sdram2region2enable WO 0x0	fpga2sdram2region1enable WO 0x0	fpga2sdram2region0enable WO 0x0	fpga2sdram1region3enable WO 0x0	fpga2sdram1region2enable WO 0x0	fpga2sdram1region1enable WO 0x0	fpga2sdram1region0enable WO 0x0	fpga2sdram0region3enable WO 0x0	fpga2sdram0region2enable WO 0x0	fpga2sdram0region1enable WO 0x0	fpga2sdram0region0enable WO 0x0	mpuregion3enable WO 0x0	mpuregion2enable WO 0x0	mpuregion1enable WO 0x0	mpuregion0enable WO 0x0

enable_clear Fields

Bit	Name	Description	Access	Reset
15	fpga2sdram2region3enable	FPGA2SDRAM2 Region 3 Enable Clear. Writing zero has no effect Writing one will clear the fpga2sdram2region3enable bit to zero	WO	0x0
14	fpga2sdram2region2enable	FPGA2SDRAM2 Region 2 Enable Clear. Writing zero has no effect Writing one will clear the fpga2sdram2region2enable bit to zero	WO	0x0
13	fpga2sdram2region1enable	FPGA2SDRAM2 Region 1 Enable Clear. Writing zero has no effect Writing one will clear the fpga2sdram2region1enable bit to zero	WO	0x0
12	fpga2sdram2region0enable	FPGA2SDRAM2 Region 0 Enable Clear. Writing zero has no effect Writing one will clear the fpga2sdram2region0enable bit to zero	WO	0x0
11	fpga2sdram1region3enable	FPGA2SDRAM1 Region 3 Enable Clear. Writing zero has no effect Writing one will clear the fpga2sdram1region3enable bit to zero	WO	0x0
10	fpga2sdram1region2enable	FPGA2SDRAM1 Region 2 Enable Clear. Writing zero has no effect Writing one will clear the fpga2sdram1region2enable bit to zero	WO	0x0

Bit	Name	Description	Access	Reset
9	fpga2sdram1region1enable	FPGA2SDRAM1 Region 1 Enable Clear. Writing zero has no effect Writing one will clear the fpga2sdram1region1enable bit to zero	WO	0x0
8	fpga2sdram1region0enable	FPGA2SDRAM1 Region 0 Enable Clear. Writing zero has no effect Writing one will clear the fpga2sdram1region0enable bit to zero	WO	0x0
7	fpga2sdram0region3enable	FPGA2SDRAM0 Region 3 Enable Clear. Writing zero has no effect Writing one will clear the fpga2sdram0region3enable bit to zero	WO	0x0
6	fpga2sdram0region2enable	FPGA2SDRAM0 Region 2 Enable Clear. Writing zero has no effect Writing one will clear the fpga2sdram0region2enable bit to zero	WO	0x0
5	fpga2sdram0region1enable	FPGA2SDRAM0 Region 1 Enable Clear. Writing zero has no effect Writing one will clear the fpga2sdram0region1enable bit to zero	WO	0x0
4	fpga2sdram0region0enable	FPGA2SDRAM0 Region 0 Enable Clear. Writing zero has no effect Writing one will clear the fpga2sdram0region0enable bit to zero	WO	0x0
3	mpuregion3enable	MPU Region 3 Enable Clear. Writing zero has no effect Writing one will clear the mpuregion3enable bit to zero	WO	0x0
2	mpuregion2enable	MPU Region 2 Enable Clear. Writing zero has no effect Writing one will clear the mpuregion2enable bit to zero	WO	0x0
1	mpuregion1enable	MPU Region 1 Enable Clear. Writing zero has no effect Writing one will clear the mpuregion1enable bit to zero	WO	0x0
0	mpuregion0enable	MPU Region 0 Enable Clear. Writing zero has no effect Writing one will clear the mpuregion0enable bit to zero	WO	0x0

mpuregion0addr

Base and Limit definition for MPU Region 0

Module Instance	Base Address	Register Address
noc_fw_ddr_mpu_fpga2sdram_ddr_scr	0xFFD13300	0xFFD13310

Offset: 0x10

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
limit RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
base RW 0x0															

mpuregion0addr Fields

Bit	Name	Description	Access	Reset
31:16	limit	Limit defines the 16 bit MSB of the address field. Remaining LSB field is all ones. Region end address is {limit, 16'hFFF}	RW	0x0
15:0	base	Base defines the 16 bit MSB of the address field. Remaining LSB field is all zeros. Region start address is {base, 16'h000}	RW	0x0

mpuregion1addr

Base and Limit definition for MPU Region 1

Module Instance	Base Address	Register Address
noc_fw_ddr_mpu_fpga2sdram_ddr_scr	0xFFD13300	0xFFD13314

Offset: 0x14

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
limit RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
base RW 0x0															

mpuregion1addr Fields

Bit	Name	Description	Access	Reset
31:16	limit	Limit defines the 16 bit MSB of the address field. Remaining LSB field is all ones. Region end address is {limit, 16'hFFF}	RW	0x0
15:0	base	Base defines the 16 bit MSB of the address field. Remaining LSB field is all zeros. Region start address is {base, 16'h000}	RW	0x0

mpuregion2addr

Base and Limit definition for MPU Region 2

Module Instance	Base Address	Register Address
noc_fw_ddr_mpu_fpga2sdram_ddr_scr	0xFFD13300	0xFFD13318

Offset: 0x18

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
limit RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
base RW 0x0															

mpuregion2addr Fields

Bit	Name	Description	Access	Reset
31:16	limit	Limit defines the 16 bit MSB of the address field. Remaining LSB field is all ones. Region end address is {limit, 16'hFFF}	RW	0x0
15:0	base	Base defines the 16 bit MSB of the address field. Remaining LSB field is all zeros. Region start address is {base, 16'h000}	RW	0x0

mpuregion3addr

Base and Limit definition for MPU Region 3

Module Instance	Base Address	Register Address
noc_fw_ddr_mpu_fpga2sdram_ddr_scr	0xFFD13300	0xFFD1331C

Offset: 0x1C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
limit RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
base RW 0x0															

mpuregion3addr Fields

Bit	Name	Description	Access	Reset
31:16	limit	Limit defines the 16 bit MSB of the address field. Remaining LSB field is all ones. Region end address is {limit, 16'hFFF}	RW	0x0
15:0	base	Base defines the 16 bit MSB of the address field. Remaining LSB field is all zeros. Region start address is {base, 16'h000}	RW	0x0

fpga2sdram0region0addr

Base and Limit definition for FPGA2SDRAM0 Region 0

Module Instance	Base Address	Register Address
noc_fw_ddr_mpu_fpga2sdram_ddr_scr	0xFFD13300	0xFFD13320

Offset: 0x20

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
limit RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
base RW 0x0															

fpga2sdram0region0addr Fields

Bit	Name	Description	Access	Reset
31:16	limit	Limit defines the 16 bit MSB of the address field. Remaining LSB field is all ones. Region end address is {limit, 16'hFFF}	RW	0x0

Bit	Name	Description	Access	Reset
15:0	base	Base defines the 16 bit MSB of the address field. Remaining LSB field is all zeros. Region start address is {base, 16'h000}	RW	0x0

fpga2sdram0region1addr

Base and Limit definition for FPGA2SDRAM0 Region 1

Module Instance	Base Address	Register Address
noc_fw_ddr_mpu_fpga2sdram_ddr_scr	0xFFD13300	0xFFD13324

Offset: 0x24

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
limit RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
base RW 0x0															

fpga2sdram0region1addr Fields

Bit	Name	Description	Access	Reset
31:16	limit	Limit defines the 16 bit MSB of the address field. Remaining LSB field is all ones. Region end address is {limit, 16'hFFF}	RW	0x0
15:0	base	Base defines the 16 bit MSB of the address field. Remaining LSB field is all zeros. Region start address is {base, 16'h000}	RW	0x0

fpga2sdram0region2addr

Base and Limit definition for FPGA2SDRAM0 Region 2

Module Instance	Base Address	Register Address
noc_fw_ddr_mpu_fpga2sdram_ddr_scr	0xFFD13300	0xFFD13328

Offset: 0x28

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
limit RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
base RW 0x0															

fpga2sdram0region2addr Fields

Bit	Name	Description	Access	Reset
31:16	limit	Limit defines the 16 bit MSB of the address field. Remaining LSB field is all ones. Region end address is {limit, 16'hFFF}	RW	0x0
15:0	base	Base defines the 16 bit MSB of the address field. Remaining LSB field is all zeros. Region start address is {base, 16'h000}	RW	0x0

fpga2sdram0region3addr

Base and Limit definition for FPGA2SDRAM0 Region 3

Module Instance	Base Address	Register Address
noc_fw_ddr_mpu_fpga2sdram_ddr_scr	0xFFD13300	0xFFD1332C

Offset: 0x2C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
limit RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
base RW 0x0															

fpga2sdram0region3addr Fields

Bit	Name	Description	Access	Reset
31:16	limit	Limit defines the 16 bit MSB of the address field. Remaining LSB field is all ones. Region end address is {limit, 16'hFFF}	RW	0x0
15:0	base	Base defines the 16 bit MSB of the address field. Remaining LSB field is all zeros. Region start address is {base, 16'h000}	RW	0x0

fpga2sdram1region0addr

Base and Limit definition for FPGA2SDRAM1 Region 0

Module Instance	Base Address	Register Address
noc_fw_ddr_mpu_fpga2sdram_ddr_scr	0xFFD13300	0xFFD13330

Offset: 0x30

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
limit RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
base RW 0x0															

fpga2sdram1region0addr Fields

Bit	Name	Description	Access	Reset
31:16	limit	Limit defines the 16 bit MSB of the address field. Remaining LSB field is all ones. Region end address is {limit, 16'hFFF}	RW	0x0
15:0	base	Base defines the 16 bit MSB of the address field. Remaining LSB field is all zeros. Region start address is {base, 16'h000}	RW	0x0

fpga2sdram1region1addr

Base and Limit definition for FPGA2SDRAM1 Region 1

Module Instance	Base Address	Register Address
noc_fw_ddr_mpu_fpga2sdram_ddr_scr	0xFFD13300	0xFFD13334

Offset: 0x34

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
limit RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
base RW 0x0															

fpga2sdram1region1addr Fields

Bit	Name	Description	Access	Reset
31:16	limit	Limit defines the 16 bit MSB of the address field. Remaining LSB field is all ones. Region end address is {limit, 16'hFFF}	RW	0x0
15:0	base	Base defines the 16 bit MSB of the address field. Remaining LSB field is all zeros. Region start address is {base, 16'h000}	RW	0x0

fpga2sdram1region2addr

Base and Limit definition for FPGA2SDRAM1 Region 2

Module Instance	Base Address	Register Address
noc_fw_ddr_mpu_fpga2sdram_ddr_scr	0xFFD13300	0xFFD13338

Offset: 0x38

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
limit RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
base RW 0x0															

fpga2sdram1region2addr Fields

Bit	Name	Description	Access	Reset
31:16	limit	Limit defines the 16 bit MSB of the address field. Remaining LSB field is all ones. Region end address is {limit, 16'hFFF}	RW	0x0

Bit	Name	Description	Access	Reset
15:0	base	Base defines the 16 bit MSB of the address field. Remaining LSB field is all zeros. Region start address is {base, 16'h000}	RW	0x0

fpga2sdram1region3addr

Base and Limit definition for FPGA2SDRAM1 Region 3

Module Instance	Base Address	Register Address
noc_fw_ddr_mpu_fpga2sdram_ddr_scr	0xFFD13300	0xFFD1333C

Offset: 0x3C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
limit RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
base RW 0x0															

fpga2sdram1region3addr Fields

Bit	Name	Description	Access	Reset
31:16	limit	Limit defines the 16 bit MSB of the address field. Remaining LSB field is all ones. Region end address is {limit, 16'hFFF}	RW	0x0
15:0	base	Base defines the 16 bit MSB of the address field. Remaining LSB field is all zeros. Region start address is {base, 16'h000}	RW	0x0

fpga2sdram2region0addr

Base and Limit definition for FPGA2SDRAM2 Region 0

Module Instance	Base Address	Register Address
noc_fw_ddr_mpu_fpga2sdram_ddr_scr	0xFFD13300	0xFFD13340

Offset: 0x40

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
limit RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
base RW 0x0															

fpga2sdram2region0addr Fields

Bit	Name	Description	Access	Reset
31:16	limit	Limit defines the 16 bit MSB of the address field. Remaining LSB field is all ones. Region end address is {limit, 16'hFFF}	RW	0x0
15:0	base	Base defines the 16 bit MSB of the address field. Remaining LSB field is all zeros. Region start address is {base, 16'h000}	RW	0x0

fpga2sdram2region1addr

Base and Limit definition for FPGA2SDRAM2 Region 1

Module Instance	Base Address	Register Address
noc_fw_ddr_mpu_fpga2sdram_ddr_scr	0xFFD13300	0xFFD13344

Offset: 0x44

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
limit RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
base RW 0x0															

fpga2sdram2region1addr Fields

Bit	Name	Description	Access	Reset
31:16	limit	Limit defines the 16 bit MSB of the address field. Remaining LSB field is all ones. Region end address is {limit, 16'hFFF}	RW	0x0
15:0	base	Base defines the 16 bit MSB of the address field. Remaining LSB field is all zeros. Region start address is {base, 16'h000}	RW	0x0

fpga2sdram2region2addr

Base and Limit definition for FPGA2SDRAM2 Region 2

Module Instance	Base Address	Register Address
noc_fw_ddr_mpu_fpga2sdram_ddr_scr	0xFFD13300	0xFFD13348

Offset: 0x48

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
limit RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
base RW 0x0															

fpga2sdram2region2addr Fields

Bit	Name	Description	Access	Reset
31:16	limit	Limit defines the 16 bit MSB of the address field. Remaining LSB field is all ones. Region end address is {limit, 16'hFFF}	RW	0x0
15:0	base	Base defines the 16 bit MSB of the address field. Remaining LSB field is all zeros. Region start address is {base, 16'h000}	RW	0x0

fpga2sdram2region3addr

Base and Limit definition for FPGA2SDRAM2 Region 3

Module Instance	Base Address	Register Address
noc_fw_ddr_mpu_fpga2sdram_ddr_scr	0xFFD13300	0xFFD1334C

Offset: 0x4C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
limit RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
base RW 0x0															

fpga2sdram2region3addr Fields

Bit	Name	Description	Access	Reset
31:16	limit	Limit defines the 16 bit MSB of the address field. Remaining LSB field is all ones. Region end address is {limit, 16'hFFF}	RW	0x0
15:0	base	Base defines the 16 bit MSB of the address field. Remaining LSB field is all zeros. Region start address is {base, 16'h000}	RW	0x0

noc_fw_ddr_l3_ddr_scr Address Map

Module Instance	Base Address	End Address
noc_fw_ddr_l3_ddr_scr	0xFFD13400	0xFFD134FF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
enable on page 7-453	0x0	32	RW	0x0	Enable
enable_set on page 7-454	0x4	32	WO	0x0	Sets Master Region Enable field when written with 1
enable_clear on page 7-456	0x8	32	WO	0x0	Clears Master Region Enable field when written with 1
hpsregion0addr on page 7-457	0xC	32	RW	0x0	Base and Limit definition for HPS Region 0
hpsregion1addr on page 7-458	0x10	32	RW	0x0	Base and Limit definition for HPS Region 1

Register	Offset	Width	Access	Reset Value	Description
hpsregion2addr on page 7-459	0x14	32	RW	0x0	Base and Limit definition for HPS Region 2
hpsregion3addr on page 7-460	0x18	32	RW	0x0	Base and Limit definition for HPS Region 3
hpsregion4addr on page 7-461	0x1C	32	RW	0x0	Base and Limit definition for HPS Region 4
hpsregion5addr on page 7-461	0x20	32	RW	0x0	Base and Limit definition for HPS Region 5
hpsregion6addr on page 7-462	0x24	32	RW	0x0	Base and Limit definition for HPS Region 6
hpsregion7addr on page 7-463	0x28	32	RW	0x0	Base and Limit definition for HPS Region 7
global on page 7-464	0x2C	32	RW	0x0	Global Firewall Control Register. This register will store various overrides that change default firewall behavior on the entire interconnect.

noc_fw_ddr_l3_ddr_scr Summary

Base Address: 0xFFD13400

Register Address Offset	Bit Fields
noc_fw_ddr_l3_ddr_scr	

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
enable 0x0	Reserved								hpsregion7enable RW 0x0	hpsregion6enable RW 0x0	hpsregion5enable RW 0x0	hpsregion4enable RW 0x0	hpsregion3enable RW 0x0	hpsregion2enable RW 0x0	hpsregion1enable RW 0x0	hpsregion0enable RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
enable_set 0x4	Reserved								hpsregion7enable WO 0x0	hpsregion6enable WO 0x0	hpsregion5enable WO 0x0	hpsregion4enable WO 0x0	hpsregion3enable WO 0x0	hpsregion2enable WO 0x0	hpsregion1enable WO 0x0	hpsregion0enable WO 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
enable_clear 0x8	Reserved								hpsregion7enable WO 0x0	hpsregion6enable WO 0x0	hpsregion5enable WO 0x0	hpsregion4enable WO 0x0	hpsregion3enable WO 0x0	hpsregion2enable WO 0x0	hpsregion1enable WO 0x0	hpsregion0enable WO 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	limit RW 0x0															
hpsregion0addr 0xC	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	base RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	limit RW 0x0															
hpsregion1addr 0x10	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	base RW 0x0															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hpsregion2addr 0x14	limit RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	base RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hpsregion3addr 0x18	limit RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	base RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hpsregion4addr 0x1C	limit RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	base RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hpsregion5addr 0x20	limit RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	base RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hpsregion6addr 0x24	limit RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	base RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hpsregion7addr 0x28	limit RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	base RW 0x0															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
global 0x2C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															error_response RW 0x0

enable

Enable

Module Instance	Base Address	Register Address
noc_fw_ddr_l3_ddr_scr	0xFFD13400	0xFFD13400

Offset: 0x0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								hpsregion7enable RW 0x0	hpsregion6enable RW 0x0	hpsregion5enable RW 0x0	hpsregion4enable RW 0x0	hpsregion3enable RW 0x0	hpsregion2enable RW 0x0	hpsregion1enable RW 0x0	hpsregion0enable RW 0x0

enable Fields

Bit	Name	Description	Access	Reset
7	hpsregion7enable	HPS Region 7 Enable. Value of 1 means region is enabled, Value of 0 means region is disabled	RW	0x0

Bit	Name	Description	Access	Reset
6	hpsregion6enable	HPS Region 6 Enable. Value of 1 means region is enabled, Value of 0 means region is disabled	RW	0x0
5	hpsregion5enable	HPS Region 5 Enable. Value of 1 means region is enabled, Value of 0 means region is disabled	RW	0x0
4	hpsregion4enable	HPS Region 4 Enable. Value of 1 means region is enabled, Value of 0 means region is disabled	RW	0x0
3	hpsregion3enable	HPS Region 3 Enable. Value of 1 means region is enabled, Value of 0 means region is disabled	RW	0x0
2	hpsregion2enable	HPS Region 2 Enable. Value of 1 means region is enabled, Value of 0 means region is disabled	RW	0x0
1	hpsregion1enable	HPS Region 1 Enable. Value of 1 means region is enabled, Value of 0 means region is disabled	RW	0x0
0	hpsregion0enable	HPS Region 0 Enable. Value of 1 means region is enabled, Value of 0 means region is disabled.	RW	0x0

enable_set

Sets Master Region Enable field when written with 1

Module Instance	Base Address	Register Address
noc_fw_ddr_l3_ddr_scr	0xFFD13400	0xFFD13404

Offset: 0x4

Access: WO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								hpsregion7enable WO 0x0	hpsregion6enable WO 0x0	hpsregion5enable WO 0x0	hpsregion4enable WO 0x0	hpsregion3enable WO 0x0	hpsregion2enable WO 0x0	hpsregion1enable WO 0x0	hpsregion0enable WO 0x0

enable_set Fields

Bit	Name	Description	Access	Reset
7	hpsregion7enable	HPS Region 0 Enable Set. Writing zero has no effect Writing one will set the hpsregion7enable bit to one	WO	0x0
6	hpsregion6enable	HPS Region 0 Enable Set. Writing zero has no effect Writing one will set the hpsregion6enable bit to one	WO	0x0
5	hpsregion5enable	HPS Region 0 Enable Set. Writing zero has no effect Writing one will set the hpsregion5enable bit to one	WO	0x0
4	hpsregion4enable	HPS Region 0 Enable Set. Writing zero has no effect Writing one will set the hpsregion4enable bit to one	WO	0x0
3	hpsregion3enable	HPS Region 0 Enable Set. Writing zero has no effect Writing one will set the hpsregion3enable bit to one	WO	0x0
2	hpsregion2enable	HPS Region 0 Enable Set. Writing zero has no effect Writing one will set the hpsregion2enable bit to one	WO	0x0
1	hpsregion1enable	HPS Region 0 Enable Set. Writing zero has no effect Writing one will set the hpsregion1enable bit to one	WO	0x0

Bit	Name	Description	Access	Reset
0	hpsregion0enable	HPS Region 0 Enable Set. Writing zero has no effect Writing one will set the hpsregion0enable bit to one	WO	0x0

enable_clear

Clears Master Region Enable field when written with 1

Module Instance	Base Address	Register Address
noc_fw_ddr_l3_ddr_scr	0xFFD13400	0xFFD13408

Offset: 0x8

Access: WO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								hpsregion7enable WO 0x0	hpsregion6enable WO 0x0	hpsregion5enable WO 0x0	hpsregion4enable WO 0x0	hpsregion3enable WO 0x0	hpsregion2enable WO 0x0	hpsregion1enable WO 0x0	hpsregion0enable WO 0x0

enable_clear Fields

Bit	Name	Description	Access	Reset
7	hpsregion7enable	HPS Region 0 Enable Clear. Writing zero has no effect Writing one will clear the hpsregion7enable bit to zero	WO	0x0
6	hpsregion6enable	HPS Region 0 Enable Clear. Writing zero has no effect Writing one will clear the hpsregion6enable bit to zero	WO	0x0

Bit	Name	Description	Access	Reset
5	hpsregion5enable	HPS Region 0 Enable Clear. Writing zero has no effect Writing one will clear the hpsregion5enable bit to zero	WO	0x0
4	hpsregion4enable	HPS Region 0 Enable Clear. Writing zero has no effect Writing one will clear the hpsregion4enable bit to zero	WO	0x0
3	hpsregion3enable	HPS Region 0 Enable Clear. Writing zero has no effect Writing one will clear the hpsregion3enable bit to zero	WO	0x0
2	hpsregion2enable	HPS Region 0 Enable Clear. Writing zero has no effect Writing one will clear the hpsregion2enable bit to zero	WO	0x0
1	hpsregion1enable	HPS Region 0 Enable Clear. Writing zero has no effect Writing one will clear the hpsregion1enable bit to zero	WO	0x0
0	hpsregion0enable	HPS Region 0 Enable Clear. Writing zero has no effect Writing one will clear the hpsregion0enable bit to zero	WO	0x0

hpsregion0addr

Base and Limit definition for HPS Region 0

Module Instance	Base Address	Register Address
noc_fw_ddr_l3_ddr_scr	0xFFD13400	0xFFD1340C

Offset: 0xC

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
limit RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
base RW 0x0															

hpsregion0addr Fields

Bit	Name	Description	Access	Reset
31:16	limit	Limit defines the 16 bit MSB of the address field. Remaining LSB field is all ones. Region end address is {limit, 16'hFFF}	RW	0x0
15:0	base	Base defines the 16 bit MSB of the address field. Remaining LSB field is all zeros. Region start address is {base, 16'h000}	RW	0x0

hpsregion1addr

Base and Limit definition for HPS Region 1

Module Instance	Base Address	Register Address
noc_fw_ddr_l3_ddr_scr	0xFFD13400	0xFFD13410

Offset: 0x10

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
limit RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
base RW 0x0															

hpsregion1addr Fields

Bit	Name	Description	Access	Reset
31:16	limit	Limit defines the 16 bit MSB of the address field. Remaining LSB field is all ones. Region end address is {limit, 16'hFFF}	RW	0x0
15:0	base	Base defines the 16 bit MSB of the address field. Remaining LSB field is all zeros. Region start address is {base, 16'h000}	RW	0x0

hpsregion2addr

Base and Limit definition for HPS Region 2

Module Instance	Base Address	Register Address
noc_fw_ddr_l3_ddr_scr	0xFFD13400	0xFFD13414

Offset: 0x14

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
limit RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
base RW 0x0															

hpsregion2addr Fields

Bit	Name	Description	Access	Reset
31:16	limit	Limit defines the 16 bit MSB of the address field. Remaining LSB field is all ones. Region end address is {limit, 16'hFFF}	RW	0x0

Bit	Name	Description	Access	Reset
15:0	base	Base defines the 16 bit MSB of the address field. Remaining LSB field is all zeros. Region start address is {base, 16'h000}	RW	0x0

hpsregion3addr

Base and Limit definition for HPS Region 3

Module Instance	Base Address	Register Address
noc_fw_ddr_l3_ddr_scr	0xFFD13400	0xFFD13418

Offset: 0x18

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
limit RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
base RW 0x0															

hpsregion3addr Fields

Bit	Name	Description	Access	Reset
31:16	limit	Limit defines the 16 bit MSB of the address field. Remaining LSB field is all ones. Region end address is {limit, 16'hFFF}	RW	0x0
15:0	base	Base defines the 16 bit MSB of the address field. Remaining LSB field is all zeros. Region start address is {base, 16'h000}	RW	0x0

hpsregion4addr

Base and Limit definition for HPS Region 4

Module Instance	Base Address	Register Address
noc_fw_ddr_l3_ddr_scr	0xFFD13400	0xFFD1341C

Offset: 0x1C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
limit RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
base RW 0x0															

hpsregion4addr Fields

Bit	Name	Description	Access	Reset
31:16	limit	Limit defines the 16 bit MSB of the address field. Remaining LSB field is all ones. Region end address is {limit, 16'hFFF}	RW	0x0
15:0	base	Base defines the 16 bit MSB of the address field. Remaining LSB field is all zeros. Region start address is {base, 16'h000}	RW	0x0

hpsregion5addr

Base and Limit definition for HPS Region 5

Module Instance	Base Address	Register Address
noc_fw_ddr_l3_ddr_scr	0xFFD13400	0xFFD13420

Offset: 0x20

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
limit RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
base RW 0x0															

hpsregion5addr Fields

Bit	Name	Description	Access	Reset
31:16	limit	Limit defines the 16 bit MSB of the address field. Remaining LSB field is all ones. Region end address is {limit, 16'hFFF}	RW	0x0
15:0	base	Base defines the 16 bit MSB of the address field. Remaining LSB field is all zeros. Region start address is {base, 16'h000}	RW	0x0

hpsregion6addr

Base and Limit definition for HPS Region 6

Module Instance	Base Address	Register Address
noc_fw_ddr_l3_ddr_scr	0xFFD13400	0xFFD13424

Offset: 0x24

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
limit RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
base RW 0x0															

hpsregion6addr Fields

Bit	Name	Description	Access	Reset
31:16	limit	Limit defines the 16 bit MSB of the address field. Remaining LSB field is all ones. Region end address is {limit, 16'hFFF}	RW	0x0
15:0	base	Base defines the 16 bit MSB of the address field. Remaining LSB field is all zeros. Region start address is {base, 16'h000}	RW	0x0

hpsregion7addr

Base and Limit definition for HPS Region 7

Module Instance	Base Address	Register Address
noc_fw_ddr_l3_ddr_scr	0xFFD13400	0xFFD13428

Offset: 0x28

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
limit RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
base RW 0x0															

hpsregion7addr Fields

Bit	Name	Description	Access	Reset
31:16	limit	Limit defines the 16 bit MSB of the address field. Remaining LSB field is all ones. Region end address is {limit, 16'hFFF}	RW	0x0

Bit	Name	Description	Access	Reset
15:0	base	Base defines the 16 bit MSB of the address field. Remaining LSB field is all zeros. Region start address is {base, 16'h000}	RW	0x0

global

Global Firewall Control Register. This register will store various overrides that change default firewall behavior on the entire interconnect.

Module Instance	Base Address	Register Address
noc_fw_ddr_l3_ddr_scr	0xFFD13400	0xFFD1342C

Offset: 0x2C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															error_response RW 0x0

global Fields

Bit	Name	Description	Access	Reset
0	error_response	When 0, transactions blocked by the firewall will silently fail (i.e successful completion with random data). When 1, transactions blocked by the firewall will return error	RW	0x0

noc_fw_soc2fpga_soc2fpga_scr Address Map

Module Instance	Base Address	End Address
noc_fw_soc2fpga_soc2fpga_scr	0xFFD13500	0xFFD13FFF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
lwsoc2fpga on page 7-466	0x0	32	RW	0x0	Per-Master Security bit for Lightweight SOC2FPGA
soc2fpga on page 7-468	0x4	32	RW	0x0	Per-Master Security bit for SOC2FPGA

noc_fw_soc2fpga_soc2fpga_scr Summary

Base Address: 0xFFD13500

Register Address Offset	Bit Fields															
noc_fw_soc2fpga_soc2fpga_scr																
lwsoc2fpga 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved						etr RW 0x0	ahb_ap RW 0x0	nand RW 0x0	sdmmc RW 0x0	usb1 RW 0x0	usb0 RW 0x0	emac2 RW 0x0	emac1 RW 0x0	emac0 RW 0x0	Reserved
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0
soc2fpga 0x4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved						etr RW 0x0	ahb_ap RW 0x0	nand RW 0x0	sdmmc RW 0x0	usb1 RW 0x0	usb0 RW 0x0	emac2 RW 0x0	emac1 RW 0x0	emac0 RW 0x0	Reserved
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0

lwsoc2fpga

Per-Master Security bit for Lightweight SOC2FPGA

Module Instance	Base Address	Register Address
noc_fw_soc2fpga_soc2fpga_scr	0xFFD13500	0xFFD13500

Offset: 0x0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved						etr RW 0x0	ahb_ap RW 0x0	nand RW 0x0	sdmmc RW 0x0	usb1 RW 0x0	usb0 RW 0x0	emac2 RW 0x0	emac1 RW 0x0	emac0 RW 0x0	Reserved
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0

lwsoc2fpga Fields

Bit	Name	Description	Access	Reset
25	etr	Security bit configuration for transactions from etr to lwsoc2fpga. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
24	ahb_ap	Security bit configuration for transactions from ahb_ap to lwsoc2fpga. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

Bit	Name	Description	Access	Reset
23	nand	Security bit configuration for transactions from nand to lwsoc2fpga. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
22	sdmmc	Security bit configuration for transactions from sdmmc to lwsoc2fpga. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
21	usb1	Security bit configuration for transactions from usb1 to lwsoc2fpga. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
20	usb0	Security bit configuration for transactions from usb0 to lwsoc2fpga. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
19	emac2	Security bit configuration for transactions from emac2 to lwsoc2fpga. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
18	emac1	Security bit configuration for transactions from emac1 to lwsoc2fpga. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
17	emac0	Security bit configuration for transactions from emac0 to lwsoc2fpga. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

Bit	Name	Description	Access	Reset
8	dma	Security bit configuration for transactions from dma to lwsoc2fpga. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
0	mpu_m0	Security bit configuration for transactions from mpu0 to lwsoc2fpga. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

soc2fpga

Per-Master Security bit for SOC2FPGA

Module Instance	Base Address	Register Address
noc_fw_soc2fpga_soc2fpga_scr	0xFFD13500	0xFFD13504

Offset: 0x4

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved						etr RW 0x0	ahb_ap RW 0x0	nand RW 0x0	sdmmc RW 0x0	usb1 RW 0x0	usb0 RW 0x0	emac2 RW 0x0	emac1 RW 0x0	emac0 RW 0x0	Reserved
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							dma RW 0x0	Reserved							mpu_m0 RW 0x0

soc2fpga Fields

Bit	Name	Description	Access	Reset
25	etr	Security bit configuration for transactions from etr to soc2fpga. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
24	ahb_ap	Security bit configuration for transactions from ahb_ap to soc2fpga. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
23	nand	Security bit configuration for transactions from nand to soc2fpga. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
22	sdmmc	Security bit configuration for transactions from sdmmc to soc2fpga. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
21	usb1	Security bit configuration for transactions from usb1 to soc2fpga. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
20	usb0	Security bit configuration for transactions from usb0 to soc2fpga. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
19	emac2	Security bit configuration for transactions from emac2 to soc2fpga. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

Bit	Name	Description	Access	Reset
18	emac1	Security bit configuration for transactions from emac1 to soc2fpga. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
17	emac0	Security bit configuration for transactions from emac0 to soc2fpga. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
8	dma	Security bit configuration for transactions from dma to soc2fpga. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0
0	mpu_m0	Security bit configuration for transactions from mpu0 to soc2fpga. When cleared (0), only Secure transactions are allowed. When set (1), both Secure and Non-Secure transactions are allowed.	RW	0x0

noc_mpu_m0_Probe_SoC2FPGA_main_Probe Address Map

Module Instance	Base Address	End Address
i_noc_mpu_m0_Probe_SoC2FPGA_main_Probe	0xFFD14000	0xFFD143FF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
Probe_SoC2FPGA_main_Probe_Id_CoreId on page 7-484	0x0	32	RO	0x567D606	

Register	Offset	Width	Access	Reset Value	Description
Probe_SoC2FPGA_main_Probe_Id_RevisionId on page 7-485	0x4	32	RO	0x129FF00	
Probe_SoC2FPGA_main_Probe_MainCtl on page 7-486	0x8	32	RW	0x0	Register MainCtl contains probe global control bits. The register has seven bit fields:
Probe_SoC2FPGA_main_Probe_CfgCtl on page 7-488	0xC	32	RW	0x0	
Probe_SoC2FPGA_main_Probe_TracePortSel on page 7-489	0x10	32	RW	0x0	
Probe_SoC2FPGA_main_Probe_FilterLut on page 7-490	0x14	32	RW	0x0	
Probe_SoC2FPGA_main_Probe_TraceAlarmEn on page 7-490	0x18	32	RW	0x0	
Probe_SoC2FPGA_main_Probe_TraceAlarmStatus on page 7-491	0x1C	32	RO	0x0	
Probe_SoC2FPGA_main_Probe_TraceAlarmClr on page 7-492	0x20	32	RW	0x0	
Probe_SoC2FPGA_main_Probe_StatPeriod on page 7-493	0x24	32	RW	0x0	
Probe_SoC2FPGA_main_Probe_StatGo on page 7-494	0x28	32	RW	0x0	
Probe_SoC2FPGA_main_Probe_StatAlarmMin on page 7-495	0x2C	32	RW	0x0	
Probe_SoC2FPGA_main_Probe_StatAlarmMax on page 7-496	0x30	32	RW	0x0	

Register	Offset	Width	Access	Reset Value	Description
Probe_SoC2FPGA_main_Probe_StatAlarmStatus on page 7-497	0x34	32	RO	0x0	
Probe_SoC2FPGA_main_Probe_StatAlarmClr on page 7-498	0x38	32	RW	0x0	
Probe_SoC2FPGA_main_Probe_StatAlarmEn on page 7-499	0x3C	32	RW	0x1	
Probe_SoC2FPGA_main_Probe_Filters_0_RouteIdBase on page 7-500	0x44	32	RW	0x0	
Probe_SoC2FPGA_main_Probe_Filters_0_RouteIdMask on page 7-501	0x48	32	RW	0x0	
Probe_SoC2FPGA_main_Probe_Filters_0_AddrBase_Low on page 7-502	0x4C	32	RW	0x0	
Probe_SoC2FPGA_main_Probe_Filters_0_WindowSize on page 7-502	0x54	32	RW	0x0	
Probe_SoC2FPGA_main_Probe_Filters_0_SecurityBase on page 7-503	0x58	32	RW	0x0	
Probe_SoC2FPGA_main_Probe_Filters_0_SecurityMask on page 7-504	0x5C	32	RW	0x0	
Probe_SoC2FPGA_main_Probe_Filters_0_Opcode on page 7-505	0x60	32	RW	0x0	Packet Probe register Opcode is a 4-bit register that selects candidate packets based on packet opcodes (0 disables the filter):
Probe_SoC2FPGA_main_Probe_Filters_0_Status on page 7-506	0x64	32	RW	0x0	Register Status is 2-bit register that selects candidate packets based on packet status.

Register	Offset	Width	Access	Reset Value	Description
Probe_SoC2FPGA_main_Probe_Filters_0_Length on page 7-507	0x68	32	RW	0x0	
Probe_SoC2FPGA_main_Probe_Filters_0_Urgency on page 7-507	0x6C	32	RW	0x0	
Probe_SoC2FPGA_main_Probe_Filters_1_RouteIdBase on page 7-508	0x80	32	RW	0x0	
Probe_SoC2FPGA_main_Probe_Filters_1_RouteIdMask on page 7-509	0x84	32	RW	0x0	
Probe_SoC2FPGA_main_Probe_Filters_1_AddrBase_Low on page 7-510	0x88	32	RW	0x0	
Probe_SoC2FPGA_main_Probe_Filters_1_WindowSize on page 7-510	0x90	32	RW	0x0	
Probe_SoC2FPGA_main_Probe_Filters_1_SecurityBase on page 7-511	0x94	32	RW	0x0	
Probe_SoC2FPGA_main_Probe_Filters_1_SecurityMask on page 7-512	0x98	32	RW	0x0	
Probe_SoC2FPGA_main_Probe_Filters_1_Opcode on page 7-513	0x9C	32	RW	0x0	Packet Probe register Opcode is a 4-bit register that selects candidate packets based on packet opcodes (0 disables the filter):
Probe_SoC2FPGA_main_Probe_Filters_1_Status on page 7-514	0xA0	32	RW	0x0	Register Status is 2-bit register that selects candidate packets based on packet status.
Probe_SoC2FPGA_main_Probe_Filters_1_Length on page 7-515	0xA4	32	RW	0x0	

Register	Offset	Width	Access	Reset Value	Description
Probe_SoC2FPGA_main_Probe_Filters_1_Urgency on page 7-516	0xA8	32	RW	0x0	
Probe_SoC2FPGA_main_Probe_Counters_0_PortSel on page 7-517	0x134	32	RW	0x0	
Probe_SoC2FPGA_main_Probe_Counters_0_Src on page 7-517	0x138	32	RW	0x0	Register CntSrc indicates the event source used to increment the counter. Unassigned values (non-existing Press level or ExtEvent index, or unimplemented Filter) are equivalent to OFF.
Probe_SoC2FPGA_main_Probe_Counters_0_AlarmMode on page 7-518	0x13C	32	RW	0x0	
Probe_SoC2FPGA_main_Probe_Counters_0_Val on page 7-519	0x140	32	RO	0x0	
Probe_SoC2FPGA_main_Probe_Counters_1_PortSel on page 7-520	0x148	32	RW	0x0	
Probe_SoC2FPGA_main_Probe_Counters_1_Src on page 7-521	0x14C	32	RW	0x0	Register CntSrc indicates the event source used to increment the counter. Unassigned values (non-existing Press level or ExtEvent index, or unimplemented Filter) are equivalent to OFF.
Probe_SoC2FPGA_main_Probe_Counters_1_AlarmMode on page 7-522	0x150	32	RW	0x0	
Probe_SoC2FPGA_main_Probe_Counters_1_Val on page 7-522	0x154	32	RO	0x0	

Register	Offset	Width	Access	Reset Value	Description
Probe_SoC2FPGA_main_Probe_Counters_2_PortSel on page 7-523	0x15C	32	RW	0x0	
Probe_SoC2FPGA_main_Probe_Counters_2_Src on page 7-524	0x160	32	RW	0x0	Register CntSrc indicates the event source used to increment the counter. Unassigned values (non-existing Press level or ExtEvent index, or unimplemented Filter) are equivalent to OFF.
Probe_SoC2FPGA_main_Probe_Counters_2_AlarmMode on page 7-525	0x164	32	RW	0x0	
Probe_SoC2FPGA_main_Probe_Counters_2_Val on page 7-526	0x168	32	RO	0x0	
Probe_SoC2FPGA_main_Probe_Counters_3_PortSel on page 7-527	0x170	32	RW	0x0	
Probe_SoC2FPGA_main_Probe_Counters_3_Src on page 7-528	0x174	32	RW	0x0	Register CntSrc indicates the event source used to increment the counter. Unassigned values (non-existing Press level or ExtEvent index, or unimplemented Filter) are equivalent to OFF.
Probe_SoC2FPGA_main_Probe_Counters_3_AlarmMode on page 7-529	0x178	32	RW	0x0	
Probe_SoC2FPGA_main_Probe_Counters_3_Val on page 7-529	0x17C	32	RO	0x0	

noc_mpu_m0_Probe_SoC2FPGA_main_Probe Summary

Base Address: 0xFFD14000

Register Address Offset	Bit Fields															
i_noc_mpu_m0_Probe_SoC2FPGA_main_Probe																
Probe_SoC2FPGA_main_Probe_Id_CoreId 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	CORECHECKSUM RO 0x567D6															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CORECHECKSUM RO 0x567D6								CORETYPEID RO 0x6							
Probe_SoC2FPGA_main_Probe_Id_RevisionId 0x4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	FLEXNOCID RO 0x129FF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FLEXNOCID RO 0x129FF								USERID RO 0x0							
Probe_SoC2FPGA_main_Probe_MainCtl 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								FILTBYTEALWAYSCHAINABLEEN RW 0x0	INTRUSIVEMODE RO 0x0	STATCONDDUMP RW 0x0	ALARMEN RW 0x0	STATEN RW 0x0	PAYLOADEN RW 0x0	TRACEEN RW 0x0	ERREN RW 0x0
Probe_SoC2FPGA_main_Probe_CfgCtl 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														ACTIVE RO 0x0	GLOBALEN RW 0x0

Register Address Offset	Bit Fields															
Probe_SoC2FPGA_main_Probe_TracePortSel 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															TRACEPORTSEL RW 0x0
Probe_SoC2FPGA_main_Probe_FilterLut 0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved												FILTERLUT RW 0x0			
Probe_SoC2FPGA_main_Probe_TraceAlarmEn 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													TRACEALARMEN RW 0x0		
Probe_SoC2FPGA_main_Probe_TraceAlarmStatus 0x1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													TRACEALARMSTATUS RO 0x0		
Probe_SoC2FPGA_main_Probe_TraceAlarmClr 0x20	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													TRACEALARMCLR RW 0x0		
Probe_SoC2FPGA_main_Probe_StatPeriod 0x24	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved											STATPERIOD RW 0x0				
Probe_SoC2FPGA_main_Probe_StatGo 0x28	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															STATGO RW 0x0

Register Address Offset	Bit Fields															
Probe_SoC2FPGA_main_Probe_StatAlarmMin 0x2C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	STATALARMMIN RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	STATALARMMIN RW 0x0															
Probe_SoC2FPGA_main_Probe_StatAlarmMax 0x30	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	STATALARMMAX RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	STATALARMMAX RW 0x0															
Probe_SoC2FPGA_main_Probe_StatAlarmStatus 0x34	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															STATALARMSTATUS RO 0x0
Probe_SoC2FPGA_main_Probe_StatAlarmClr 0x38	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															STATALARMCLR RW 0x0
Probe_SoC2FPGA_main_Probe_StatAlarmEn 0x3C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															STATALARMEN RW 0x1
Probe_SoC2FPGA_main_Probe_Filters_0_RouteId-Base 0x44	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved													FILTERS_0_ROUTEID-BASE RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FILTERS_0_ROUTEIDBASE RW 0x0															

Register Address Offset	Bit Fields															
Probe_SoC2FPGA_main_Probe_Filters_0_RouteId-Mask 0x48	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved													FILTERS_0_ROUTEID-MASK RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FILTERS_0_ROUTEIDMASK RW 0x0															
Probe_SoC2FPGA_main_Probe_Filters_0_AddrBase_Low 0x4C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	FILTERS_0_ADDRBASE_LOW RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FILTERS_0_ADDRBASE_LOW RW 0x0															
Probe_SoC2FPGA_main_Probe_Filters_0_WindowSize 0x54	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved										FILTERS_0_WINDOWSIZE RW 0x0					
Probe_SoC2FPGA_main_Probe_Filters_0_Security Base 0x58	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													FILTERS_0_SECURITY-BASE RW 0x0		
Probe_SoC2FPGA_main_Probe_Filters_0_Security Mask 0x5C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													FILTERS_0_SECURITY-MASK RW 0x0		
Probe_SoC2FPGA_main_Probe_Filters_0_Opcode 0x60	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved												URGEN RW 0x0	LOCKEN RW 0x0	WREN RW 0x0	RDEN RW 0x0

Register Address Offset	Bit Fields															
Probe_SoC2FPGA_main_Probe_Filters_0_Status 0x64	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														RSPEN RW 0x0	REQEN RW 0x0
Probe_SoC2FPGA_main_Probe_Filters_0_Length 0x68	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved												FILTERS_0_LENGTH RW 0x0			
Probe_SoC2FPGA_main_Probe_Filters_0_Urgency 0x6C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														FILTERS_0_URGENCY RW 0x0	
Probe_SoC2FPGA_main_Probe_Filters_1_RouteId-Base 0x80	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved													FILTERS_1_ROUTEID-BASE RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FILTERS_1_ROUTEIDBASE RW 0x0															
Probe_SoC2FPGA_main_Probe_Filters_1_RouteId-Mask 0x84	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved													FILTERS_1_ROUTEID-MASK RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FILTERS_1_ROUTEIDMASK RW 0x0															
Probe_SoC2FPGA_main_Probe_Filters_1_AddrBase_Low 0x88	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	FILTERS_1_ADDRBASE_LOW RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FILTERS_1_ADDRBASE_LOW RW 0x0															

Register Address Offset	Bit Fields															
Probe_SoC2FPGA_main_Probe_Filters_1_WindowSize 0x90	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved										FILTERS_1_WINDOWSIZE RW 0x0					
Probe_SoC2FPGA_main_Probe_Filters_1_SecurityBase 0x94	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													FILTERS_1_SECURITY-BASE RW 0x0		
Probe_SoC2FPGA_main_Probe_Filters_1_SecurityMask 0x98	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													FILTERS_1_SECURITY-MASK RW 0x0		
Probe_SoC2FPGA_main_Probe_Filters_1_Opcode 0x9C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved												URGEN RW 0x0	LOCKEN RW 0x0	WREN RW 0x0	RDEN RW 0x0
Probe_SoC2FPGA_main_Probe_Filters_1_Status 0xA0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														RSPEN RW 0x0	REQEN RW 0x0
Probe_SoC2FPGA_main_Probe_Filters_1_Length 0xA4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved												FILTERS_1_LENGTH RW 0x0			

Register Address Offset	Bit Fields															
Probe_SoC2FPGA_main_Probe_Filters_1_Urgency 0xA8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														FILTERS_1_URGENCY RW 0x0	
Probe_SoC2FPGA_main_Probe_Counters_0_PortSel 0x134	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															COUNTERS_0_PORTSEL RW 0x0
Probe_SoC2FPGA_main_Probe_Counters_0_Src 0x138	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved											INTEVENT RW 0x0				
Probe_SoC2FPGA_main_Probe_Counters_0_AlarmMode 0x13C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														COUNTERS_0_ALARMMODE RW 0x0	
Probe_SoC2FPGA_main_Probe_Counters_0_Val 0x140	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	COUNTERS_0_VAL RO 0x0															
Probe_SoC2FPGA_main_Probe_Counters_1_PortSel 0x148	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															COUNTERS_1_PORTSEL RW 0x0

Register Address Offset	Bit Fields															
Probe_SoC2FPGA_main_Probe_Counters_1_Src 0x14C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved											INTEVENT RW 0x0				
Probe_SoC2FPGA_main_Probe_Counters_1_AlarmMode 0x150	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														COUNTERS_1_ALARMMODE RW 0x0	
Probe_SoC2FPGA_main_Probe_Counters_1_Val 0x154	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	COUNTERS_1_VAL RO 0x0															
Probe_SoC2FPGA_main_Probe_Counters_2_PortSel 0x15C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															COUNTERS_2_PORTSEL RW 0x0
Probe_SoC2FPGA_main_Probe_Counters_2_Src 0x160	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved											INTEVENT RW 0x0				
Probe_SoC2FPGA_main_Probe_Counters_2_AlarmMode 0x164	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														COUNTERS_2_ALARMMODE RW 0x0	

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Probe_SoC2FPGA_main_Probe_Counters_2_Val 0x168	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	COUNTERS_2_VAL RO 0x0															
Probe_SoC2FPGA_main_Probe_Counters_3_PortSel 0x170	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															COUNTERS_3_PORTSEL RW 0x0
Probe_SoC2FPGA_main_Probe_Counters_3_Src 0x174	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved											INTEVENT RW 0x0				
Probe_SoC2FPGA_main_Probe_Counters_3_AlarmMode 0x178	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														COUNTERS_3_ALARMMODE RW 0x0	
Probe_SoC2FPGA_main_Probe_Counters_3_Val 0x17C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	COUNTERS_3_VAL RO 0x0															

Probe_SoC2FPGA_main_Probe_Id_CoreId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_SoC2FPGA_main_Probe	0xFFD14000	0xFFD14000

Offset: 0x0

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CORECHECKSUM RO 0x567D6															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CORECHECKSUM RO 0x567D6								CORETYPEID RO 0x6							

Probe_SoC2FPGA_main_Probe_Id_CoreId Fields

Bit	Name	Description	Access	Reset
31:8	CORECHECKSUM	Field containing a checksum of the parameters of the IP.	RO	0x567D6
7:0	CORETYPEID	Field identifying the type of IP.	RO	0x6

Probe_SoC2FPGA_main_Probe_Id_RevisionId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_SoC2FPGA_main_Probe	0xFFD14000	0xFFD14004

Offset: 0x4

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLEXNOCID RO 0x129FF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLEXNOCID RO 0x129FF								USERID RO 0x0							

Probe_SoC2FPGA_main_Probe_Id_RevisionId Fields

Bit	Name	Description	Access	Reset
31:8	FLEXNOCID	Field containing the build revision of the software used to generate the IP HDL code.	RO	0x129FF
7:0	USERID	Field containing a user defined value, not used anywhere inside the IP itself.	RO	0x0

Probe_SoC2FPGA_main_Probe_MainCtl

Register MainCtl contains probe global control bits. The register has seven bit fields:

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_SoC2FPGA_main_Probe	0xFFD14000	0xFFD14008

Offset: 0x8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								FILTBYTEALWAYSCHAINABLEEN RW 0x0	INTRUSIVEMODE RO 0x0	STATCONDDUMP RW 0x0	ALARMEN RW 0x0	STATEN RW 0x0	PAYLOADEN RW 0x0	TRACEEN RW 0x0	ERREN RW 0x0

Probe_SoC2FPGA_main_Probe_MainCtl Fields

Bit	Name	Description	Access	Reset
7	FILTBYTEALWAYSCHAINABLEEN	When set to 0, filters are mapped to all statistic counters when counting bytes or enabled bytes. Therefore, only filter events mapped to even counters can be counted using a pair of chained counters.When set to 1, filters are mapped only to even statistic counters when counting bytes or enabled bytes. Thus events from any filter can be counted using a pair of chained counters.	RW	0x0
6	INTRUSIVEMODE	When set to 1, register field IntrusiveMode enables trace operation in Intrusive flow-control mode. When set to 0, the register enables trace operation in Overflow flow-control mode	RO	0x0
5	STATCONDDUMP	When set, register field StatCond-Dump enables the dump of a statistics frame to the range of counter values set for registers StatAlarmMin, StatAlarmMax, and AlarmMode. This field also renders register StatAlarmStatus inopera-tive. When parameter statisticsCoun-terAlarm is set to False, the StatCondDump register bit is reserved.	RW	0x0
4	ALARMEN	When set, register field AlarmEn enables the probe to collect alarm-related information. When the register field bit is null, both TraceAlarm and StatAlarm outputs are driven to 0.	RW	0x0
3	STATEN	When set to 1, register field StatEn enables statistics profiling. The probe sendS statistics results to the output for signal ObsTx. All statistics counters are cleared when the StatEn bit goes from 0 to 1. When set to 0, counters are disabled.	RW	0x0

Bit	Name	Description	Access	Reset
2	PAYLOADEN	Register field PayloadEn, when set to 1, enables traces to contain headers and payload. When set ot 0, only headers are reported.	RW	0x0
1	TRACEEN	Register field TraceEn enables the probe to send filtered packets (Trace) on the ObsTx observation output.	RW	0x0
0	ERREN	Register field ErrEn enables the probe to send on the ObsTx output any packet with Error status, independently of filtering mechanisms, thus constituting a simple supplementary global filter.	RW	0x0

Probe_SoC2FPGA_main_Probe_CfgCtl

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_SoC2FPGA_main_Probe	0xFFD14000	0xFFD1400C

Offset: 0xC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														ACTIVE RO 0x0	GLOBALEN RW 0x0

Probe_SoC2FPGA_main_Probe_CfgCtl Fields

Bit	Name	Description	Access	Reset
1	ACTIVE		RO	0x0
0	GLOBALEN		RW	0x0

Probe_SoC2FPGA_main_Probe_TracePortSel

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_SoC2FPGA_main_Probe	0xFFD14000	0xFFD14010

Offset: 0x10

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															TRACEPORTSEL RW 0x0

Probe_SoC2FPGA_main_Probe_TracePortSel Fields

Bit	Name	Description	Access	Reset
0	TRACEPORTSEL	Register TracePortSel indicates which generic protocol link is currently being observed by trace logic.The number of bits in register TracePortSel is equal to log2 of the value set for parameter nPort.The register can be updated at any time, but changes only become effective at packet boundaries.	RW	0x0

Probe_SoC2FPGA_main_Probe_FilterLut

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_SoC2FPGA_main_Probe	0xFFD14000	0xFFD14014

Offset: 0x14

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												FILTERLUT RW 0x0			

Probe_SoC2FPGA_main_Probe_FilterLut Fields

Bit	Name	Description	Access	Reset
3:0	FILTERLUT	Register FilterLut contains a look-up table that is used to combine filter outputs in order to trace packets. Packet tracing is enabled when the FilterLut bit of index (FNout ... F0out) is equal to 1.The number of bits in register FilterLut is determined by the setting for parameter nFilter, calculated as 2**nFilter.When parameter nFilter is set to None, FilterLut is reserved.	RW	0x0

Probe_SoC2FPGA_main_Probe_TraceAlarmEn

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_SoC2FPGA_main_Probe	0xFFD14000	0xFFD14018

Offset: 0x18

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													TRACEALARMEN RW 0x0		

Probe_SoC2FPGA_main_Probe_TraceAlarmEn Fields

Bit	Name	Description	Access	Reset
2:0	TRACEALARMEN	Register TraceAlarmEn controls which lookup table or filter can set the TraceAlarm signal output once the trace alarm status is set. The number of bits in register TraceAlarmEn is determined by the value set for parameter nFilter + 1.Bit nFilter controls the lookup table output, and bits nFilter:0 control the corresponding filter output. When parameter nFilter is set to None, TraceAlarmEn is reserved.	RW	0x0

Probe_SoC2FPGA_main_Probe_TraceAlarmStatus

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_SoC2FPGA_main_Probe	0xFFD14000	0xFFD1401C

Offset: 0x1C

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													TRACEALARMSTATUS RO 0x0		

Probe_SoC2FPGA_main_Probe_TraceAlarmStatus Fields

Bit	Name	Description	Access	Reset
2:0	TRACEALARMSTATUS	Register TraceAlarmStatus is a read-only register that indicates which lookup table or filter has been matched by a packet, independently of register TraceAlarmEn bit configuration. The number of bits in TraceAlarmStatus is determined by the value set for parameter nFilter + 1.When nFilter is set to None, TraceAlarmStatus is reserved.	RO	0x0

Probe_SoC2FPGA_main_Probe_TraceAlarmClr

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_SoC2FPGA_main_Probe	0xFFD14000	0xFFD14020

Offset: 0x20

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													TRACEALARMCLR RW 0x0		

Probe_SoC2FPGA_main_Probe_TraceAlarmClr Fields

Bit	Name	Description	Access	Reset
2:0	TRACEALARMCLR	Setting a bit to 1 in register TraceAlarmClr clears the corresponding bit in register TraceAlarmStatus.The number of bits in register TraceAlarmClr is equal to (nFilter + 1). When nFilter is set to 0, TraceAlarmClr is reserved.NOTE The written value is not stored in TraceAlarmClr. A read always returns 0.	RW	0x0

Probe_SoC2FPGA_main_Probe_StatPeriod

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_SoC2FPGA_main_Probe	0xFFD14000	0xFFD14024

Offset: 0x24

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved											STATPERIOD RW 0x0				

Probe_SoC2FPGA_main_Probe_StatPeriod Fields

Bit	Name	Description	Access	Reset
4:0	STATPERIOD	Register StatPeriod is a 5-bit register that sets a period, within a range of 2 cycles to 2 gigacycles, during which statistics are collected before being dumped automatically. Setting the register implicitly enables automatic mode operation for statistics collection. The period is calculated with the formula: N_Cycle = 2**StatPeriodWhen register StatPeriod is set to its default value 0, automatic dump mode is disabled, and register StatGo is activated for manual mode operation. Note: When parameter statisticsCollection is set to False, StatPeriod is reserved.	RW	0x0

Probe_SoC2FPGA_main_Probe_StatGo

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_SoC2FPGA_main_Probe	0xFFD14000	0xFFD14028

Offset: 0x28

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															STATGO RW 0x0

Probe_SoC2FPGA_main_Probe_StatGo Fields

Bit	Name	Description	Access	Reset
0	STATGO	Writing a 1 to the 1-bit pulse register StatGo generates a statistics dump.The register is active when statistics collection operates in manual mode, that is, when register StatPeriod is set to 0.NOTE The written value is not stored in StatGo. A read always returns 0.	RW	0x0

Probe_SoC2FPGA_main_Probe_StatAlarmMin

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_SoC2FPGA_main_Probe	0xFFD14000	0xFFD1402C

Offset: 0x2C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
STATALARMMIN RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STATALARMMIN RW 0x0															

Probe_SoC2FPGA_main_Probe_StatAlarmMin Fields

Bit	Name	Description	Access	Reset
31:0	STATALARMMIN	Register StatAlarmMin contains the minimum count value used in statistics alarm comparisons. The number of bits is equal to twice the value set forparameter wStatisticsCounter. When parameter statisticsCounterAlarm is set to False, StatAlarmMin is reserved.	RW	0x0

Probe_SoC2FPGA_main_Probe_StatAlarmMax

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_SoC2FPGA_main_Probe	0xFFD14000	0xFFD14030

Offset: 0x30

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
STATALARMMAX RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STATALARMMAX RW 0x0															

Probe_SoC2FPGA_main_Probe_StatAlarmMax Fields

Bit	Name	Description	Access	Reset
31:0	STATALARMMAX	Register StatAlarmMax contains the maximum count value used in statistics alarm comparisons.The number of bits is equal to twice the value set for parameter wStatisticsCounter. When parameter statisticsCounterAlarm is set to False, StatAlarmMax is reserved.	RW	0x0

Probe_SoC2FPGA_main_Probe_StatAlarmStatus

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_SoC2FPGA_main_Probe	0xFFD14000	0xFFD14034

Offset: 0x34

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															STATALARMSTATUS RO 0x0

Probe_SoC2FPGA_main_Probe_StatAlarmStatus Fields

Bit	Name	Description	Access	Reset
0	STATALARMSTATUS	Register StatAlarmStatus is a read-only 1-bit register indicating that at least one statistics counter has exceeded the programmed values for registers StatAlarmMin or StatAlarmMax. Output signal StatAlarm is equal to the values stored in register MainCtl fields StatAlarmStatus and AlarmEn. When parameter statisticsCounterAlarm is set to False, StatAlarmStatus is reserved.	RO	0x0

Probe_SoC2FPGA_main_Probe_StatAlarmClr

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_SoC2FPGA_main_Probe	0xFFD14000	0xFFD14038

Offset: 0x38

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															STATALARMCLR RW 0x0

Probe_SoC2FPGA_main_Probe_StatAlarmClr Fields

Bit	Name	Description	Access	Reset
0	STATALARMCLR	Register StatAlarmClr is a 1-bit register. Writing a 1 to this register clears the StatAlarmStatus register bit.When parameter statisticsCounterAlarm is set to False, StatAlarmClr is reserved.NOTE The written value is not stored in StatAlarmClr. A read always returns 0.	RW	0x0

Probe_SoC2FPGA_main_Probe_StatAlarmEn

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_SoC2FPGA_main_Probe	0xFFD14000	0xFFD1403C

Offset: 0x3C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															STATALARMEN RW 0x1

Probe_SoC2FPGA_main_Probe_StatAlarmEn Fields

Bit	Name	Description	Access	Reset
0	STATALARMEN	Register StatAlarmEn is a 1-bit register. When set to 0 it masks StatAlarm and CtiTrigOut(1) signal interrupts.	RW	0x1

Probe_SoC2FPGA_main_Probe_Filters_0_RouteIdBase

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_SoC2FPGA_main_Probe	0xFFD14000	0xFFD14044

Offset: 0x44

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													FILTERS_0_ROUTEIDBASE RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FILTERS_0_ROUTEIDBASE RW 0x0															

Probe_SoC2FPGA_main_Probe_Filters_0_RouteIdBase Fields

Bit	Name	Description	Access	Reset
18:0	FILTERS_0_ROUTEIDBASE	Register RouteIdBase contains the RouteId-lsbFilterRouteId bits base used to filter packets.	RW	0x0

Probe_SoC2FPGA_main_Probe_Filters_0_RouteIdMask

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_SoC2FPGA_main_Probe	0xFFD14000	0xFFD14048

Offset: 0x48

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													FILTERS_0_ROUTEIDMASK RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FILTERS_0_ROUTEIDMASK RW 0x0															

Probe_SoC2FPGA_main_Probe_Filters_0_RouteIdMask Fields

Bit	Name	Description	Access	Reset
18:0	FILTERS_0_ROUTEIDMASK	Register RouteIdMask contains the RouteId-lsbFilterRouteId mask used to filter packets. A packet is a candidate when packet.RouteId>> lsbFilterRouteId & RouteIdMask = RouteIdBase & RouteIdMask.	RW	0x0

Probe_SoC2FPGA_main_Probe_Filters_0_AddrBase_Low

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_SoC2FPGA_main_Probe	0xFFD14000	0xFFD1404C

Offset: 0x4C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FILTERS_0_ADDRBASE_LOW RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FILTERS_0_ADDRBASE_LOW RW 0x0															

Probe_SoC2FPGA_main_Probe_Filters_0_AddrBase_Low Fields

Bit	Name	Description	Access	Reset
31:0	FILTERS_0_ADDRBASE_LOW	Address LSB register.	RW	0x0

Probe_SoC2FPGA_main_Probe_Filters_0_WindowSize

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_SoC2FPGA_main_Probe	0xFFD14000	0xFFD14054

Offset: 0x54

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										FILTERS_0_WINDOWSIZE RW 0x0					

Probe_SoC2FPGA_main_Probe_Filters_0_WindowSize Fields

Bit	Name	Description	Access	Reset
5:0	FILTERS_0_WINDOWSIZE	Register WindowSize contains the encoded address mask used to filter packets. The effective Mask value is equal to ~(2max(WindowSize, packet.Len) - 1). A packet is a candidate when packet.Addr & Mask = AddrBase & Mask. This allows filteringof packets having an intersection with the AddrBase/ WindowSize burst aligned region, even if the region is smaller than the packet.	RW	0x0

Probe_SoC2FPGA_main_Probe_Filters_0_SecurityBase

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_SoC2FPGA_main_Probe	0xFFD14000	0xFFD14058

Offset: 0x58

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													FILTERS_0_SECURITY-BASE RW 0x0		

Probe_SoC2FPGA_main_Probe_Filters_0_SecurityBase Fields

Bit	Name	Description	Access	Reset
2:0	FILTERS_0_SECURITYBASE	Register SecurityBase contains the security base used to filter packets.	RW	0x0

Probe_SoC2FPGA_main_Probe_Filters_0_SecurityMask

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_SoC2FPGA_main_Probe	0xFFD14000	0xFFD1405C

Offset: 0x5C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													FILTERS_0_SECURITY-MASK RW 0x0		

Probe_SoC2FPGA_main_Probe_Filters_0_SecurityMask Fields

Bit	Name	Description	Access	Reset
2:0	FILTERS_0_SECURITYMASK	Register SecurityMask is contains the security mask used to filter packets. A packet is a candidate when: packet.Security & Security-Mask = SecurityBase & SecurityMasks.	RW	0x0

Probe_SoC2FPGA_main_Probe_Filters_0_Opcode

Packet Probe register Opcode is a 4-bit register that selects candidate packets based on packet opcodes (0 disables the filter):

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_SoC2FPGA_main_Probe	0xFFD14000	0xFFD14060

Offset: 0x60

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												URGEN RW 0x0	LOCKEN RW 0x0	WREN RW 0x0	RDEN RW 0x0

Probe_SoC2FPGA_main_Probe_Filters_0_Opcode Fields

Bit	Name	Description	Access	Reset
3	URGEN	Selects URG packets (urgency).	RW	0x0

Bit	Name	Description	Access	Reset
2	LOCKEN	Selects RDX-WR, RDL, WRC and Linked sequence.	RW	0x0
1	WREN	Selects WR packets.	RW	0x0
0	RDEN	Selects RD packets.	RW	0x0

Probe_SoC2FPGA_main_Probe_Filters_0_Status

Register Status is 2-bit register that selects candidate packets based on packet status.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_SoC2FPGA_main_Probe	0xFFD14000	0xFFD14064

Offset: 0x64

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														RSPEN RW 0x0	REQEN RW 0x0

Probe_SoC2FPGA_main_Probe_Filters_0_Status Fields

Bit	Name	Description	Access	Reset
1	RSPEN	Selects RSP and FAIL-CONT status packets.	RW	0x0
0	REQEN	Selects REQ status packets.	RW	0x0

Probe_SoC2FPGA_main_Probe_Filters_0_Length

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_SoC2FPGA_main_Probe	0xFFD14000	0xFFD14068

Offset: 0x68

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												FILTERS_0_LENGTH RW 0x0			

Probe_SoC2FPGA_main_Probe_Filters_0_Length Fields

Bit	Name	Description	Access	Reset
3:0	FILTERS_0_LENGTH	Register Length is 4-bit register that selects candidate packets if their number of bytes is less than or equal to 2**Length.	RW	0x0

Probe_SoC2FPGA_main_Probe_Filters_0_Urgency

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_SoC2FPGA_main_Probe	0xFFD14000	0xFFD1406C

Offset: 0x6C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														FILTERS_0_URGENCY RW 0x0	

Probe_SoC2FPGA_main_Probe_Filters_0_Urgency Fields

Bit	Name	Description	Access	Reset
1:0	FILTERS_0_URGENCY	Register Urgency contains the minimum urgency level used to filter packets. A packet is a candidate when its socket urgency is greater than or equal to the urgency specified in the register.	RW	0x0

Probe_SoC2FPGA_main_Probe_Filters_1_RouteIdBase

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_SoC2FPGA_main_Probe	0xFFD14000	0xFFD14080

Offset: 0x80

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													FILTERS_1_ROUTEIDBASE RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FILTERS_1_ROUTEIDBASE RW 0x0															

Probe_SoC2FPGA_main_Probe_Filters_1_RouteIdBase Fields

Bit	Name	Description	Access	Reset
18:0	FILTERS_1_ROUTEIDBASE	Register RouteIdBase contains the RouteId-lsbFilterRouteId bits base used to filter packets.	RW	0x0

Probe_SoC2FPGA_main_Probe_Filters_1_RouteIdMask

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_SoC2FPGA_main_Probe	0xFFD14000	0xFFD14084

Offset: 0x84

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													FILTERS_1_ROUTEIDMASK RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FILTERS_1_ROUTEIDMASK RW 0x0															

Probe_SoC2FPGA_main_Probe_Filters_1_RouteIdMask Fields

Bit	Name	Description	Access	Reset
18:0	FILTERS_1_ROUTEIDMASK	Register RouteIdMask contains the RouteId-lsbFilterRouteId mask used to filter packets. A packet is a candidate when packet.RouteId>> lsbFilterRouteId & RouteIdMask = RouteIdBase & RouteIdMask.	RW	0x0

Probe_SoC2FPGA_main_Probe_Filters_1_AddrBase_Low

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_SoC2FPGA_main_Probe	0xFFD14000	0xFFD14088

Offset: 0x88

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FILTERS_1_ADDRBASE_LOW RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FILTERS_1_ADDRBASE_LOW RW 0x0															

Probe_SoC2FPGA_main_Probe_Filters_1_AddrBase_Low Fields

Bit	Name	Description	Access	Reset
31:0	FILTERS_1_ADDRBASE_LOW	Address LSB register.	RW	0x0

Probe_SoC2FPGA_main_Probe_Filters_1_WindowSize

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_SoC2FPGA_main_Probe	0xFFD14000	0xFFD14090

Offset: 0x90

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										FILTERS_1_WINDOWSIZE RW 0x0					

Probe_SoC2FPGA_main_Probe_Filters_1_WindowSize Fields

Bit	Name	Description	Access	Reset
5:0	FILTERS_1_WINDOWSIZE	Register WindowSize contains the encoded address mask used to filter packets. The effective Mask value is equal to ~(2max(WindowSize, packet.Len) - 1). A packet is a candidate when packet.Addr & Mask = AddrBase & Mask. This allows filteringof packets having an intersection with the AddrBase/ WindowSize burst aligned region, even if the region is smaller than the packet.	RW	0x0

Probe_SoC2FPGA_main_Probe_Filters_1_SecurityBase

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_SoC2FPGA_main_Probe	0xFFD14000	0xFFD14094

Offset: 0x94

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													FILTERS_1_SECURITY-BASE RW 0x0		

Probe_SoC2FPGA_main_Probe_Filters_1_SecurityBase Fields

Bit	Name	Description	Access	Reset
2:0	FILTERS_1_SECURITYBASE	Register SecurityBase contains the security base used to filter packets.	RW	0x0

Probe_SoC2FPGA_main_Probe_Filters_1_SecurityMask

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_SoC2FPGA_main_Probe	0xFFD14000	0xFFD14098

Offset: 0x98

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													FILTERS_1_SECURITY-MASK RW 0x0		

Probe_SoC2FPGA_main_Probe_Filters_1_SecurityMask Fields

Bit	Name	Description	Access	Reset
2:0	FILTERS_1_SECURITYMASK	Register SecurityMask is contains the security mask used to filter packets. A packet is a candidate when: packet.Security & Security-Mask = SecurityBase & SecurityMasks.	RW	0x0

Probe_SoC2FPGA_main_Probe_Filters_1_Opcode

Packet Probe register Opcode is a 4-bit register that selects candidate packets based on packet opcodes (0 disables the filter):

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_SoC2FPGA_main_Probe	0xFFD14000	0xFFD1409C

Offset: 0x9C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												URGEN RW 0x0	LOCKEN RW 0x0	WREN RW 0x0	RDEN RW 0x0

Probe_SoC2FPGA_main_Probe_Filters_1_Opcode Fields

Bit	Name	Description	Access	Reset
3	URGEN	Selects URG packets (urgency).	RW	0x0
2	LOCKEN	Selects RDX-WR, RDL, WRC and Linked sequence.	RW	0x0
1	WREN	Selects WR packets.	RW	0x0
0	RDEN	Selects RD packets.	RW	0x0

Probe_SoC2FPGA_main_Probe_Filters_1_Status

Register Status is 2-bit register that selects candidate packets based on packet status.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_SoC2FPGA_main_Probe	0xFFD14000	0xFFD140A0

Offset: 0xA0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														RSPEN RW 0x0	REQEN RW 0x0

Probe_SoC2FPGA_main_Probe_Filters_1_Status Fields

Bit	Name	Description	Access	Reset
1	RSPEN	Selects RSP and FAIL-CONT status packets.	RW	0x0
0	REQEN	Selects REQ status packets.	RW	0x0

Probe_SoC2FPGA_main_Probe_Filters_1_Length

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_SoC2FPGA_main_Probe	0xFFD14000	0xFFD140A4

Offset: 0xA4

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												FILTERS_1_LENGTH RW 0x0			

Probe_SoC2FPGA_main_Probe_Filters_1_Length Fields

Bit	Name	Description	Access	Reset
3:0	FILTERS_1_LENGTH	Register Length is 4-bit register that selects candidate packets if their number of bytes is less than or equal to 2**Length.	RW	0x0

Probe_SoC2FPGA_main_Probe_Filters_1_Urgency

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_SoC2FPGA_main_Probe	0xFFD14000	0xFFD140A8

Offset: 0xA8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														FILTERS_1_URGENCY RW 0x0	

Probe_SoC2FPGA_main_Probe_Filters_1_Urgency Fields

Bit	Name	Description	Access	Reset
1:0	FILTERS_1_URGENCY	Register Urgency contains the minimum urgency level used to filter packets. A packet is a candidate when its socket urgency is greater than or equal to the urgency specified in the register.	RW	0x0

Probe_SoC2FPGA_main_Probe_Counters_0_PortSel

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_SoC2FPGA_main_Probe	0xFFD14000	0xFFD14134

Offset: 0x134

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															COUNTERS_0_PORTSEL RW 0x0

Probe_SoC2FPGA_main_Probe_Counters_0_PortSel Fields

Bit	Name	Description	Access	Reset
0	COUNTERS_0_PORTSEL	Register PortSel indicates which NTTP link is associated with the counter. The register can be changed at any time, with the change effective immediately. The LUT and FILTx sources do not depend on this NTTP port selection.	RW	0x0

Probe_SoC2FPGA_main_Probe_Counters_0_Src

Register CntSrc indicates the event source used to increment the counter. Unassigned values (non-existing Press level or ExtEvent index, or unimplemented Filter) are equivalent to OFF.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_SoC2FPGA_main_Probe	0xFFD14000	0xFFD14138

Offset: 0x138

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved											INTEVENT RW 0x0				

Probe_SoC2FPGA_main_Probe_Counters_0_Src Fields

Bit	Name	Description	Access	Reset
4:0	INTEVENT	Internal packet event	RW	0x0

Probe_SoC2FPGA_main_Probe_Counters_0_AlarmMode

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_SoC2FPGA_main_Probe	0xFFD14000	0xFFD1413C

Offset: 0x13C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														COUNTERS_0_ALARMMODE RW 0x0	

Probe_SoC2FPGA_main_Probe_Counters_0_AlarmMode Fields

Bit	Name	Description	Access	Reset
1:0	COUNTERS_0_ALARMMODE	Register AlarmMode is a 2-bit register that is present when parameter statisticsCounterAlarm is set to True. The register defines the statistics-alarm behavior of the counter.	RW	0x0

Probe_SoC2FPGA_main_Probe_Counters_0_Val

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_SoC2FPGA_main_Probe	0xFFD14000	0xFFD14140

Offset: 0x140

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COUNTERS_0_VAL RO 0x0															

Probe_SoC2FPGA_main_Probe_Counters_0_Val Fields

Bit	Name	Description	Access	Reset
15:0	COUNTERS_0_VAL	Register Val is a read-only register that is always present. The register containsthe statistics counter value either pending StatAlarm output, or when statisticscollection is suspended subsequent to triggers or signal statSuspend.	RO	0x0

Probe_SoC2FPGA_main_Probe_Counters_1_PortSel

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_SoC2FPGA_main_Probe	0xFFD14000	0xFFD14148

Offset: 0x148

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															COUNTERS_1_PORTSEL RW 0x0

Probe_SoC2FPGA_main_Probe_Counters_1_PortSel Fields

Bit	Name	Description	Access	Reset
0	COUNTERS_1_PORTSEL	Register PortSel indicates which NTTP link is associated with the counter. The register can be changed at any time, with the change effective immediately. The LUT and FILTx sources do not depend on this NTTP port selection.	RW	0x0

Probe_SoC2FPGA_main_Probe_Counters_1_Src

Register CntSrc indicates the event source used to increment the counter. Unassigned values (non-existing Press level or ExtEvent index, or unimplemented Filter) are equivalent to OFF.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_SoC2FPGA_main_Probe	0xFFD14000	0xFFD1414C

Offset: 0x14C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved											INTEVENT RW 0x0				

Probe_SoC2FPGA_main_Probe_Counters_1_Src Fields

Bit	Name	Description	Access	Reset
4:0	INTEVENT	Internal packet event	RW	0x0

Probe_SoC2FPGA_main_Probe_Counters_1_AlarmMode

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_SoC2FPGA_main_Probe	0xFFD14000	0xFFD14150

Offset: 0x150

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														COUNTERS_1_ALARMMODE RW 0x0	

Probe_SoC2FPGA_main_Probe_Counters_1_AlarmMode Fields

Bit	Name	Description	Access	Reset
1:0	COUNTERS_1_ALARMMODE	Register AlarmMode is a 2-bit register that is present when parameter statisticsCounterAlarm is set to True. The register defines the statistics-alarm behavior of the counter.	RW	0x0

Probe_SoC2FPGA_main_Probe_Counters_1_Val

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_SoC2FPGA_main_Probe	0xFFD14000	0xFFD14154

Offset: 0x154

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COUNTERS_1_VAL RO 0x0															

Probe_SoC2FPGA_main_Probe_Counters_1_Val Fields

Bit	Name	Description	Access	Reset
15:0	COUNTERS_1_VAL	Register Val is a read-only register that is always present. The register containsthe statistics counter value either pending StatAlarm output, or when statisticscollection is suspended subsequent to triggers or signal statSuspend.	RO	0x0

Probe_SoC2FPGA_main_Probe_Counters_2_PortSel

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_SoC2FPGA_main_Probe	0xFFD14000	0xFFD1415C

Offset: 0x15C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															COUNTERS_2_PORTSEL RW 0x0

Probe_SoC2FPGA_main_Probe_Counters_2_PortSel Fields

Bit	Name	Description	Access	Reset
0	COUNTERS_2_PORTSEL	Register PortSel indicates which NTTP link is associated with the counter. The register can be changed at any time, with the change effective immediately. The LUT and FILTx sources do not depend on this NTTP port selection.	RW	0x0

Probe_SoC2FPGA_main_Probe_Counters_2_Src

Register CntSrc indicates the event source used to increment the counter. Unassigned values (non-existing Press level or ExtEvent index, or unimplemented Filter) are equivalent to OFF.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_SoC2FPGA_main_Probe	0xFFD14000	0xFFD14160

Offset: 0x160

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved											INTEVENT RW 0x0				

Probe_SoC2FPGA_main_Probe_Counters_2_Src Fields

Bit	Name	Description	Access	Reset
4:0	INTEVENT	Internal packet event	RW	0x0

Probe_SoC2FPGA_main_Probe_Counters_2_AlarmMode

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_SoC2FPGA_main_Probe	0xFFD14000	0xFFD14164

Offset: 0x164

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														COUNTERS_2_ALARMMODE RW 0x0	

Probe_SoC2FPGA_main_Probe_Counters_2_AlarmMode Fields

Bit	Name	Description	Access	Reset
1:0	COUNTERS_2_ALARMMODE	Register AlarmMode is a 2-bit register that is present when parameter statisticsCounterAlarm is set to True. The register defines the statistics-alarm behavior of the counter.	RW	0x0

Probe_SoC2FPGA_main_Probe_Counters_2_Val

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_SoC2FPGA_main_Probe	0xFFD14000	0xFFD14168

Offset: 0x168

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COUNTERS_2_VAL RO 0x0															

Probe_SoC2FPGA_main_Probe_Counters_2_Val Fields

Bit	Name	Description	Access	Reset
15:0	COUNTERS_2_VAL	Register Val is a read-only register that is always present. The register containsthe statistics counter value either pending StatAlarm output, or when statisticscollection is suspended subsequent to triggers or signal statSuspend.	RO	0x0

Probe_SoC2FPGA_main_Probe_Counters_3_PortSel

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_SoC2FPGA_main_Probe	0xFFD14000	0xFFD14170

Offset: 0x170

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															COUNTERS_3_PORTSEL RW 0x0

Probe_SoC2FPGA_main_Probe_Counters_3_PortSel Fields

Bit	Name	Description	Access	Reset
0	COUNTERS_3_PORTSEL	Register PortSel indicates which NTTP link is associated with the counter. The register can be changed at any time, with the change effective immediately. The LUT and FILTx sources do not depend on this NTTP port selection.	RW	0x0

Probe_SoC2FPGA_main_Probe_Counters_3_Src

Register CntSrc indicates the event source used to increment the counter. Unassigned values (non-existing Press level or ExtEvent index, or unimplemented Filter) are equivalent to OFF.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_SoC2FPGA_main_Probe	0xFFD14000	0xFFD14174

Offset: 0x174

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved											INTEVENT RW 0x0				

Probe_SoC2FPGA_main_Probe_Counters_3_Src Fields

Bit	Name	Description	Access	Reset
4:0	INTEVENT	Internal packet event	RW	0x0

Probe_SoC2FPGA_main_Probe_Counters_3_AlarmMode

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_SoC2FPGA_main_Probe	0xFFD14000	0xFFD14178

Offset: 0x178

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														COUNTERS_3_ALARMMODE RW 0x0	

Probe_SoC2FPGA_main_Probe_Counters_3_AlarmMode Fields

Bit	Name	Description	Access	Reset
1:0	COUNTERS_3_ALARMMODE	Register AlarmMode is a 2-bit register that is present when parameter statisticsCounterAlarm is set to True. The register defines the statistics-alarm behavior of the counter.	RW	0x0

Probe_SoC2FPGA_main_Probe_Counters_3_Val

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_SoC2FPGA_main_Probe	0xFFD14000	0xFFD1417C

Offset: 0x17C

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COUNTERS_3_VAL RO 0x0															

Probe_SoC2FPGA_main_Probe_Counters_3_Val Fields

Bit	Name	Description	Access	Reset
15:0	COUNTERS_3_VAL	Register Val is a read-only register that is always present. The register containsthe statistics counter value either pending StatAlarm output, or when statisticscollection is suspended subsequent to triggers or signal statSuspend.	RO	0x0

noc_mpu_m0_Probe_emacs_main_Probe Address Map

Module Instance	Base Address	End Address
i_noc_mpu_m0_Probe_emacs_main_Probe	0xFFD14400	0xFFD147FF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
Probe_emacs_main_Probe_Id_CoreId on page 7-544	0x0	32	RO	0xF46F6306	

Register	Offset	Width	Access	Reset Value	Description
Probe_emacs_main_Probe_Id_RevisionId on page 7-545	0x4	32	RO	0x129FF00	
Probe_emacs_main_Probe_MainCtl on page 7-546	0x8	32	RW	0x0	Register MainCtl contains probe global control bits. The register has seven bit fields:
Probe_emacs_main_Probe_CfgCtl on page 7-548	0xC	32	RW	0x0	
Probe_emacs_main_Probe_TracePortSel on page 7-549	0x10	32	RW	0x0	
Probe_emacs_main_Probe_FilterLut on page 7-550	0x14	32	RW	0x0	
Probe_emacs_main_Probe_TraceAlarmEn on page 7-550	0x18	32	RW	0x0	
Probe_emacs_main_Probe_TraceAlarmStatus on page 7-551	0x1C	32	RO	0x0	
Probe_emacs_main_Probe_TraceAlarmClr on page 7-552	0x20	32	RW	0x0	
Probe_emacs_main_Probe_StatPeriod on page 7-553	0x24	32	RW	0x0	
Probe_emacs_main_Probe_StatGo on page 7-554	0x28	32	RW	0x0	
Probe_emacs_main_Probe_StatAlarmMin on page 7-555	0x2C	32	RW	0x0	
Probe_emacs_main_Probe_StatAlarmMax on page 7-556	0x30	32	RW	0x0	

Register	Offset	Width	Access	Reset Value	Description
Probe_emacs_main_Probe_StatAlarmStatus on page 7-557	0x34	32	RO	0x0	
Probe_emacs_main_Probe_StatAlarmClr on page 7-558	0x38	32	RW	0x0	
Probe_emacs_main_Probe_StatAlarmEn on page 7-559	0x3C	32	RW	0x1	
Probe_emacs_main_Probe_Filters_0_RouteId-Base on page 7-560	0x44	32	RW	0x0	
Probe_emacs_main_Probe_Filters_0_RouteId-Mask on page 7-561	0x48	32	RW	0x0	
Probe_emacs_main_Probe_Filters_0_AddrBase_Low on page 7-562	0x4C	32	RW	0x0	
Probe_emacs_main_Probe_Filters_0_WindowSize on page 7-562	0x54	32	RW	0x0	
Probe_emacs_main_Probe_Filters_0_Security-Base on page 7-563	0x58	32	RW	0x0	
Probe_emacs_main_Probe_Filters_0_Security-Mask on page 7-564	0x5C	32	RW	0x0	
Probe_emacs_main_Probe_Filters_0_Opcode on page 7-565	0x60	32	RW	0x0	Packet Probe register Opcode is a 4-bit register that selects candidate packets based on packet opcodes (0 disables the filter):
Probe_emacs_main_Probe_Filters_0_Status on page 7-566	0x64	32	RW	0x0	Register Status is 2-bit register that selects candidate packets based on packet status.

Register	Offset	Width	Access	Reset Value	Description
Probe_emacs_main_Probe_Filters_0_Length on page 7-567	0x68	32	RW	0x0	
Probe_emacs_main_Probe_Filters_0_Urgency on page 7-567	0x6C	32	RW	0x0	
Probe_emacs_main_Probe_Filters_1_RouteId-Base on page 7-568	0x80	32	RW	0x0	
Probe_emacs_main_Probe_Filters_1_RouteId-Mask on page 7-569	0x84	32	RW	0x0	
Probe_emacs_main_Probe_Filters_1_AddrBase_Low on page 7-570	0x88	32	RW	0x0	
Probe_emacs_main_Probe_Filters_1_WindowSize on page 7-570	0x90	32	RW	0x0	
Probe_emacs_main_Probe_Filters_1_Security-Base on page 7-571	0x94	32	RW	0x0	
Probe_emacs_main_Probe_Filters_1_Security-Mask on page 7-572	0x98	32	RW	0x0	
Probe_emacs_main_Probe_Filters_1_Opcode on page 7-573	0x9C	32	RW	0x0	Packet Probe register Opcode is a 4-bit register that selects candidate packets based on packet opcodes (0 disables the filter):
Probe_emacs_main_Probe_Filters_1_Status on page 7-574	0xA0	32	RW	0x0	Register Status is 2-bit register that selects candidate packets based on packet status.
Probe_emacs_main_Probe_Filters_1_Length on page 7-575	0xA4	32	RW	0x0	

Register	Offset	Width	Access	Reset Value	Description
Probe_emacs_main_Probe_Filters_1_Urgency on page 7-576	0xA8	32	RW	0x0	
Probe_emacs_main_Probe_Counters_0_PortSel on page 7-577	0x134	32	RW	0x0	
Probe_emacs_main_Probe_Counters_0_Src on page 7-577	0x138	32	RW	0x0	Register CntSrc indicates the event source used to increment the counter. Unassigned values (non-existing Press level or ExtEvent index, or unimplemented Filter) are equivalent to OFF.
Probe_emacs_main_Probe_Counters_0_AlarmMode on page 7-578	0x13C	32	RW	0x0	
Probe_emacs_main_Probe_Counters_0_Val on page 7-579	0x140	32	RO	0x0	
Probe_emacs_main_Probe_Counters_1_PortSel on page 7-580	0x148	32	RW	0x0	
Probe_emacs_main_Probe_Counters_1_Src on page 7-581	0x14C	32	RW	0x0	Register CntSrc indicates the event source used to increment the counter. Unassigned values (non-existing Press level or ExtEvent index, or unimplemented Filter) are equivalent to OFF.
Probe_emacs_main_Probe_Counters_1_AlarmMode on page 7-582	0x150	32	RW	0x0	
Probe_emacs_main_Probe_Counters_1_Val on page 7-583	0x154	32	RO	0x0	

Register	Offset	Width	Access	Reset Value	Description
Probe_emacs_main_Probe_Counters_2_PortSel on page 7-583	0x15C	32	RW	0x0	
Probe_emacs_main_Probe_Counters_2_Src on page 7-584	0x160	32	RW	0x0	Register CntSrc indicates the event source used to increment the counter. Unassigned values (non-existing Press level or ExtEvent index, or unimplemented Filter) are equivalent to OFF.
Probe_emacs_main_Probe_Counters_2_AlarmMode on page 7-585	0x164	32	RW	0x0	
Probe_emacs_main_Probe_Counters_2_Val on page 7-586	0x168	32	RO	0x0	
Probe_emacs_main_Probe_Counters_3_PortSel on page 7-587	0x170	32	RW	0x0	
Probe_emacs_main_Probe_Counters_3_Src on page 7-588	0x174	32	RW	0x0	Register CntSrc indicates the event source used to increment the counter. Unassigned values (non-existing Press level or ExtEvent index, or unimplemented Filter) are equivalent to OFF.
Probe_emacs_main_Probe_Counters_3_AlarmMode on page 7-589	0x178	32	RW	0x0	
Probe_emacs_main_Probe_Counters_3_Val on page 7-590	0x17C	32	RO	0x0	

noc_mpu_m0_Probe_emacs_main_Probe Summary

Base Address: 0xFFD14400

Register Address Offset	Bit Fields															
i_noc_mpu_m0_Probe_emacs_main_Probe																
Probe_emacs_main_Probe_Id_CoreId 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	CORECHECKSUM RO 0xF46F63															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CORECHECKSUM RO 0xF46F63								CORETYPEID RO 0x6							
Probe_emacs_main_Probe_Id_RevisionId 0x4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	FLEXNOCID RO 0x129FF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FLEXNOCID RO 0x129FF								USERID RO 0x0							
Probe_emacs_main_Probe_MainCtl 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								FILTBYTEALWAYSCHAINABLEEN RW 0x0	INTRUSIVEMODE RO 0x0	STATCONDDUMP RW 0x0	ALARMEN RW 0x0	STATEN RW 0x0	PAYLOADEN RW 0x0	TRACEEN RW 0x0	ERREN RW 0x0
Probe_emacs_main_Probe_CfgCtl 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														ACTIVE RO 0x0	GLOBALEN RW 0x0

Register Address Offset	Bit Fields															
Probe_emacs_main_Probe_TracePortSel 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															TRACEPORTSEL RW 0x0
Probe_emacs_main_Probe_FilterLut 0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved												FILTERLUT RW 0x0			
Probe_emacs_main_Probe_TraceAlarmEn 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													TRACEALARMEN RW 0x0		
Probe_emacs_main_Probe_TraceAlarmStatus 0x1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													TRACEALARMSTATUS RO 0x0		
Probe_emacs_main_Probe_TraceAlarmClr 0x20	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													TRACEALARMCLR RW 0x0		
Probe_emacs_main_Probe_StatPeriod 0x24	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved											STATPERIOD RW 0x0				
Probe_emacs_main_Probe_StatGo 0x28	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															STATGO RW 0x0

Register Address Offset	Bit Fields															
Probe_emacs_main_Probe_StatAlarmMin 0x2C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	STATALARMMIN RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	STATALARMMIN RW 0x0															
Probe_emacs_main_Probe_StatAlarmMax 0x30	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	STATALARMMAX RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	STATALARMMAX RW 0x0															
Probe_emacs_main_Probe_StatAlarm-Status 0x34	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															STATALARMSTATUS RO 0x0
Probe_emacs_main_Probe_StatAlarmClr 0x38	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															STATALARMCLR RW 0x0
Probe_emacs_main_Probe_StatAlarmEn 0x3C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															STATALARMEN RW 0x1
Probe_emacs_main_Probe_Filters_0_RouteIdBase 0x44	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved													FILTERS_0_ROUTEID-BASE RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FILTERS_0_ROUTEIDBASE RW 0x0															

Register Address Offset	Bit Fields															
Probe_emacs_main_Probe_Filters_0_RouteIdMask 0x48	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved													FILTERS_0_ROUTEID-MASK RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FILTERS_0_ROUTEIDMASK RW 0x0															
Probe_emacs_main_Probe_Filters_0_AddrBase_Low 0x4C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	FILTERS_0_ADDRBASE_LOW RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FILTERS_0_ADDRBASE_LOW RW 0x0															
Probe_emacs_main_Probe_Filters_0_WindowSize 0x54	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved										FILTERS_0_WINDOWSIZE RW 0x0					
Probe_emacs_main_Probe_Filters_0_Security-Base 0x58	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													FILTERS_0_SECURITY-BASE RW 0x0		
Probe_emacs_main_Probe_Filters_0_Security-Mask 0x5C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													FILTERS_0_SECURITY-MASK RW 0x0		
Probe_emacs_main_Probe_Filters_0_Opcode 0x60	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved												URGEN RW 0x0	LOCKEN RW 0x0	WREN RW 0x0	RDEN RW 0x0

Register Address Offset	Bit Fields															
Probe_emacs_main_Probe_Filters_0_Status 0x64	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														RSPEN RW 0x0	REQEN RW 0x0
Probe_emacs_main_Probe_Filters_0_Length 0x68	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved												FILTERS_0_LENGTH RW 0x0			
Probe_emacs_main_Probe_Filters_0_Urgency 0x6C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														FILTERS_0_URGENCY RW 0x0	
Probe_emacs_main_Probe_Filters_1_RouteIdBase 0x80	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved													FILTERS_1_ROUTEID-BASE RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FILTERS_1_ROUTEIDBASE RW 0x0															
Probe_emacs_main_Probe_Filters_1_RouteIdMask 0x84	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved													FILTERS_1_ROUTEID-MASK RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FILTERS_1_ROUTEIDMASK RW 0x0															
Probe_emacs_main_Probe_Filters_1_AddrBase_Low 0x88	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	FILTERS_1_ADDRBASE_LOW RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FILTERS_1_ADDRBASE_LOW RW 0x0															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Probe_emacs_main_Probe_Filters_1_WindowSize 0x90	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved										FILTERS_1_WINDOWSIZE RW 0x0					
Probe_emacs_main_Probe_Filters_1_Security-Base 0x94	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													FILTERS_1_SECURITY-BASE RW 0x0		
Probe_emacs_main_Probe_Filters_1_Security-Mask 0x98	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													FILTERS_1_SECURITY-MASK RW 0x0		
Probe_emacs_main_Probe_Filters_1_Opcode 0x9C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved												URGEN RW 0x0	LOCKEN RW 0x0	WREN RW 0x0	RDEN RW 0x0
Probe_emacs_main_Probe_Filters_1_Status 0xA0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														RSPEN RW 0x0	REQEN RW 0x0
Probe_emacs_main_Probe_Filters_1_Length 0xA4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved												FILTERS_1_LENGTH RW 0x0			

Register Address Offset	Bit Fields															
Probe_emacs_main_Probe_Filters_1_Urgency 0xA8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														FILTERS_1_URGENCY RW 0x0	
Probe_emacs_main_Probe_Counters_0_PortSel 0x134	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															COUNTERS_0_PORTSEL RW 0x0
Probe_emacs_main_Probe_Counters_0_Src 0x138	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved										EXTEVENT RW 0x0	INTEVENT RW 0x0				
Probe_emacs_main_Probe_Counters_0_AlarmMode 0x13C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														COUNTERS_0_ALARMMODE RW 0x0	
Probe_emacs_main_Probe_Counters_0_Val 0x140	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	COUNTERS_0_VAL RO 0x0															
Probe_emacs_main_Probe_Counters_1_PortSel 0x148	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															COUNTERS_1_PORTSEL RW 0x0

Register Address Offset	Bit Fields															
Probe_emacs_main_Probe_Counters_1_Src 0x14C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved										EXTEVENT RW 0x0	INTEVENT RW 0x0				
Probe_emacs_main_Probe_Counters_1_AlarmMode 0x150	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														COUNTERS_1_ALARMMODE RW 0x0	
Probe_emacs_main_Probe_Counters_1_Val 0x154	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	COUNTERS_1_VAL RO 0x0															
Probe_emacs_main_Probe_Counters_2_PortSel 0x15C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															COUNTERS_2_PORTSEL RW 0x0
Probe_emacs_main_Probe_Counters_2_Src 0x160	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved										EXTEVENT RW 0x0	INTEVENT RW 0x0				
Probe_emacs_main_Probe_Counters_2_AlarmMode 0x164	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														COUNTERS_2_ALARMMODE RW 0x0	

Register Address Offset	Bit Fields															
Probe_emacs_main_Probe_Counters_2_Val 0x168	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	COUNTERS_2_VAL RO 0x0															
Probe_emacs_main_Probe_Counters_3_PortSel 0x170	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															COUNTERS_3_PORTSEL RW 0x0
Probe_emacs_main_Probe_Counters_3_Src 0x174	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved										EXTEVENT RW 0x0	INTEVENT RW 0x0				
Probe_emacs_main_Probe_Counters_3_AlarmMode 0x178	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														COUNTERS_3_ALARMMODE RW 0x0	
Probe_emacs_main_Probe_Counters_3_Val 0x17C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	COUNTERS_3_VAL RO 0x0															

Probe_emacs_main_Probe_Id_CoreId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_emacs_main_Probe	0xFFD14400	0xFFD14400

Offset: 0x0

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CORECHECKSUM RO 0xF46F63															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CORECHECKSUM RO 0xF46F63								CORETYPEID RO 0x6							

Probe_emacs_main_Probe_Id_CoreId Fields

Bit	Name	Description	Access	Reset
31:8	CORECHECKSUM	Field containing a checksum of the parameters of the IP.	RO	0xF46F63
7:0	CORETYPEID	Field identifying the type of IP.	RO	0x6

Probe_emacs_main_Probe_Id_RevisionId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_emacs_main_Probe	0xFFD14400	0xFFD14404

Offset: 0x4

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLEXNOCID RO 0x129FF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLEXNOCID RO 0x129FF								USERID RO 0x0							

Probe_emacs_main_Probe_Id_RevisionId Fields

Bit	Name	Description	Access	Reset
31:8	FLEXNOCID	Field containing the build revision of the software used to generate the IP HDL code.	RO	0x129FF
7:0	USERID	Field containing a user defined value, not used anywhere inside the IP itself.	RO	0x0

Probe_emacs_main_Probe_MainCtl

Register MainCtl contains probe global control bits. The register has seven bit fields:

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_emacs_main_Probe	0xFFD14400	0xFFD14408

Offset: 0x8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								FILTBYTEALWAYSCHAINABLEEN RW 0x0	INTRUSIVEMODE RO 0x0	STATCONDDUMP RW 0x0	ALARMEN RW 0x0	STATEN RW 0x0	PAYLOADEN RW 0x0	TRACEEN RW 0x0	ERREN RW 0x0

Probe_emacs_main_Probe_MainCtl Fields

Bit	Name	Description	Access	Reset
7	FILTBYTEALWAYSCHAINABLEEN	When set to 0, filters are mapped to all statistic counters when counting bytes or enabled bytes. Therefore, only filter events mapped to even counters can be counted using a pair of chained counters.When set to 1, filters are mapped only to even statistic counters when counting bytes or enabled bytes. Thus events from any filter can be counted using a pair of chained counters.	RW	0x0
6	INTRUSIVEMODE	When set to 1, register field IntrusiveMode enables trace operation in Intrusive flow-control mode. When set to 0, the register enables trace operation in Overflow flow-control mode	RO	0x0
5	STATCONDDUMP	When set, register field StatCond-Dump enables the dump of a statistics frame to the range of counter values set for registers StatAlarmMin, StatAlarmMax, and AlarmMode. This field also renders register StatAlarmStatus inopera-tive. When parameter statisticsCoun-terAlarm is set to False, the StatCondDump register bit is reserved.	RW	0x0
4	ALARMEN	When set, register field AlarmEn enables the probe to collect alarm-related information. When the register field bit is null, both TraceAlarm and StatAlarm outputs are driven to 0.	RW	0x0
3	STATEN	When set to 1, register field StatEn enables statistics profiling. The probe sendS statistics results to the output for signal ObsTx. All statistics counters are cleared when the StatEn bit goes from 0 to 1. When set to 0, counters are disabled.	RW	0x0

Bit	Name	Description	Access	Reset
2	PAYLOADEN	Register field PayloadEn, when set to 1, enables traces to contain headers and payload. When set ot 0, only headers are reported.	RW	0x0
1	TRACEEN	Register field TraceEn enables the probe to send filtered packets (Trace) on the ObsTx observation output.	RW	0x0
0	ERREN	Register field ErrEn enables the probe to send on the ObsTx output any packet with Error status, independently of filtering mechanisms, thus constituting a simple supplementary global filter.	RW	0x0

Probe_emacs_main_Probe_CfgCtl

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_emacs_main_Probe	0xFFD14400	0xFFD1440C

Offset: 0xC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														ACTIVE RO 0x0	GLOBALEN RW 0x0

Probe_emacs_main_Probe_CfgCtl Fields

Bit	Name	Description	Access	Reset
1	ACTIVE		RO	0x0
0	GLOBALEN		RW	0x0

Probe_emacs_main_Probe_TracePortSel

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_emacs_main_Probe	0xFFD14400	0xFFD14410

Offset: 0x10

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															TRACEPORTSEL RW 0x0

Probe_emacs_main_Probe_TracePortSel Fields

Bit	Name	Description	Access	Reset
0	TRACEPORTSEL	Register TracePortSel indicates which generic protocol link is currently being observed by trace logic.The number of bits in register TracePortSel is equal to log2 of the value set for parameter nPort.The register can be updated at any time, but changes only become effective at packet boundaries.	RW	0x0

Probe_emacs_main_Probe_FilterLut

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_emacs_main_Probe	0xFFD14400	0xFFD14414

Offset: 0x14

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												FILTERLUT RW 0x0			

Probe_emacs_main_Probe_FilterLut Fields

Bit	Name	Description	Access	Reset
3:0	FILTERLUT	Register FilterLut contains a look-up table that is used to combine filter outputs in order to trace packets. Packet tracing is enabled when the FilterLut bit of index (FNout ... F0out) is equal to 1.The number of bits in register FilterLut is determined by the setting for parameter nFilter, calculated as 2**nFilter.When parameter nFilter is set to None, FilterLut is reserved.	RW	0x0

Probe_emacs_main_Probe_TraceAlarmEn

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_emacs_main_Probe	0xFFD14400	0xFFD14418

Offset: 0x18

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													TRACEALARMEN RW 0x0		

Probe_emacs_main_Probe_TraceAlarmEn Fields

Bit	Name	Description	Access	Reset
2:0	TRACEALARMEN	Register TraceAlarmEn controls which lookup table or filter can set the TraceAlarm signal output once the trace alarm status is set. The number of bits in register TraceAlarmEn is determined by the value set for parameter nFilter + 1.Bit nFilter controls the lookup table output, and bits nFilter:0 control the corresponding filter output. When parameter nFilter is set to None, TraceAlarmEn is reserved.	RW	0x0

Probe_emacs_main_Probe_TraceAlarmStatus

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_emacs_main_Probe	0xFFD14400	0xFFD1441C

Offset: 0x1C

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													TRACEALARMSTATUS RO 0x0		

Probe_emacs_main_Probe_TraceAlarmStatus Fields

Bit	Name	Description	Access	Reset
2:0	TRACEALARMSTATUS	Register TraceAlarmStatus is a read-only register that indicates which lookup table or filter has been matched by a packet, independently of register TraceAlarmEn bit configuration. The number of bits in TraceAlarmStatus is determined by the value set for parameter nFilter + 1.When nFilter is set to None, TraceAlarmStatus is reserved.	RO	0x0

Probe_emacs_main_Probe_TraceAlarmClr

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_emacs_main_Probe	0xFFD14400	0xFFD14420

Offset: 0x20

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													TRACEALARMCLR RW 0x0		

Probe_emacs_main_Probe_TraceAlarmClr Fields

Bit	Name	Description	Access	Reset
2:0	TRACEALARMCLR	Setting a bit to 1 in register TraceAlarmClr clears the corresponding bit in register TraceAlarmStatus.The number of bits in register TraceAlarmClr is equal to (nFilter + 1). When nFilter is set to 0, TraceAlarmClr is reserved.NOTE The written value is not stored in TraceAlarmClr. A read always returns 0.	RW	0x0

Probe_emacs_main_Probe_StatPeriod

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_emacs_main_Probe	0xFFD14400	0xFFD14424

Offset: 0x24

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved											STATPERIOD RW 0x0				

Probe_emacs_main_Probe_StatPeriod Fields

Bit	Name	Description	Access	Reset
4:0	STATPERIOD	Register StatPeriod is a 5-bit register that sets a period, within a range of 2 cycles to 2 gigacycles, during which statistics are collected before being dumped automatically. Setting the register implicitly enables automatic mode operation for statistics collection. The period is calculated with the formula: N_Cycle = 2**StatPeriodWhen register StatPeriod is set to its default value 0, automatic dump mode is disabled, and register StatGo is activated for manual mode operation. Note: When parameter statisticsCollection is set to False, StatPeriod is reserved.	RW	0x0

Probe_emacs_main_Probe_StatGo

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_emacs_main_Probe	0xFFD14400	0xFFD14428

Offset: 0x28

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															STATGO RW 0x0

Probe_emacs_main_Probe_StatGo Fields

Bit	Name	Description	Access	Reset
0	STATGO	Writing a 1 to the 1-bit pulse register StatGo generates a statistics dump.The register is active when statistics collection operates in manual mode, that is, when register StatPeriod is set to 0.NOTE The written value is not stored in StatGo. A read always returns 0.	RW	0x0

Probe_emacs_main_Probe_StatAlarmMin

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_emacs_main_Probe	0xFFD14400	0xFFD1442C

Offset: 0x2C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
STATALARMMIN RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STATALARMMIN RW 0x0															

Probe_emacs_main_Probe_StatAlarmMin Fields

Bit	Name	Description	Access	Reset
31:0	STATALARMMIN	Register StatAlarmMin contains the minimum count value used in statistics alarm comparisons. The number of bits is equal to twice the value set forparameter wStatisticsCounter. When parameter statisticsCounterAlarm is set to False, StatAlarmMin is reserved.	RW	0x0

Probe_emacs_main_Probe_StatAlarmMax

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_emacs_main_Probe	0xFFD14400	0xFFD14430

Offset: 0x30

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
STATALARMMAX RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STATALARMMAX RW 0x0															

Probe_emacs_main_Probe_StatAlarmMax Fields

Bit	Name	Description	Access	Reset
31:0	STATALARMMAX	Register StatAlarmMax contains the maximum count value used in statistics alarm comparisons.The number of bits is equal to twice the value set for parameter wStatisticsCounter. When parameter statisticsCounterAlarm is set to False, StatAlarmMax is reserved.	RW	0x0

Probe_emacs_main_Probe_StatAlarmStatus

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_emacs_main_Probe	0xFFD14400	0xFFD14434

Offset: 0x34

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															STATALARMSTATUS RO 0x0

Probe_emacs_main_Probe_StatAlarmStatus Fields

Bit	Name	Description	Access	Reset
0	STATALARMSTATUS	Register StatAlarmStatus is a read-only 1-bit register indicating that at least one statistics counter has exceeded the programmed values for registers StatAlarmMin or StatAlarmMax. Output signal StatAlarm is equal to the values stored in register MainCtl fields StatAlarmStatus and AlarmEn. When parameter statisticsCounterAlarm is set to False, StatAlarmStatus is reserved.	RO	0x0

Probe_emacs_main_Probe_StatAlarmClr

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_emacs_main_Probe	0xFFD14400	0xFFD14438

Offset: 0x38

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															STATALARMCLR RW 0x0

Probe_emacs_main_Probe_StatAlarmClr Fields

Bit	Name	Description	Access	Reset
0	STATALARMCLR	Register StatAlarmClr is a 1-bit register. Writing a 1 to this register clears the StatAlarmStatus register bit.When parameter statisticsCounterAlarm is set to False, StatAlarmClr is reserved.NOTE The written value is not stored in StatAlarmClr. A read always returns 0.	RW	0x0

Probe_emacs_main_Probe_StatAlarmEn

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_emacs_main_Probe	0xFFD14400	0xFFD1443C

Offset: 0x3C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															STATALARMEN RW 0x1

Probe_emacs_main_Probe_StatAlarmEn Fields

Bit	Name	Description	Access	Reset
0	STATALARMEN	Register StatAlarmEn is a 1-bit register. When set to 0 it masks StatAlarm and CtiTrigOut(1) signal interrupts.	RW	0x1

Probe_emacs_main_Probe_Filters_0_RouteIdBase

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_emacs_main_Probe	0xFFD14400	0xFFD14444

Offset: 0x44

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													FILTERS_0_ROUTEIDBASE RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FILTERS_0_ROUTEIDBASE RW 0x0															

Probe_emacs_main_Probe_Filters_0_RouteIdBase Fields

Bit	Name	Description	Access	Reset
18:0	FILTERS_0_ROUTEIDBASE	Register RouteIdBase contains the RouteId-lsbFilterRouteId bits base used to filter packets.	RW	0x0

Probe_emacs_main_Probe_Filters_0_RouteIdMask

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_emacs_main_Probe	0xFFD14400	0xFFD14448

Offset: 0x48

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													FILTERS_0_ROUTEIDMASK RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FILTERS_0_ROUTEIDMASK RW 0x0															

Probe_emacs_main_Probe_Filters_0_RouteIdMask Fields

Bit	Name	Description	Access	Reset
18:0	FILTERS_0_ROUTEIDMASK	Register RouteIdMask contains the RouteId-lsbFilterRouteId mask used to filter packets. A packet is a candidate when packet.RouteId>> lsbFilterRouteId & RouteIdMask = RouteIdBase & RouteIdMask.	RW	0x0

Probe_emacs_main_Probe_Filters_0_AddrBase_Low

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_emacs_main_Probe	0xFFD14400	0xFFD1444C

Offset: 0x4C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FILTERS_0_ADDRBASE_LOW RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FILTERS_0_ADDRBASE_LOW RW 0x0															

Probe_emacs_main_Probe_Filters_0_AddrBase_Low Fields

Bit	Name	Description	Access	Reset
31:0	FILTERS_0_ADDRBASE_LOW	Address LSB register.	RW	0x0

Probe_emacs_main_Probe_Filters_0_WindowSize

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_emacs_main_Probe	0xFFD14400	0xFFD14454

Offset: 0x54

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										FILTERS_0_WINDOWSIZE RW 0x0					

Probe_emacs_main_Probe_Filters_0_WindowSize Fields

Bit	Name	Description	Access	Reset
5:0	FILTERS_0_WINDOWSIZE	Register WindowSize contains the encoded address mask used to filter packets. The effective Mask value is equal to ~(2max(WindowSize, packet.Len) - 1). A packet is a candidate when packet.Addr & Mask = AddrBase & Mask. This allows filteringof packets having an intersection with the AddrBase/ WindowSize burst aligned region, even if the region is smaller than the packet.	RW	0x0

Probe_emacs_main_Probe_Filters_0_SecurityBase

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_emacs_main_Probe	0xFFD14400	0xFFD14458

Offset: 0x58

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													FILTERS_0_SECURITY-BASE RW 0x0		

Probe_emacs_main_Probe_Filters_0_SecurityBase Fields

Bit	Name	Description	Access	Reset
2:0	FILTERS_0_SECURITYBASE	Register SecurityBase contains the security base used to filter packets.	RW	0x0

Probe_emacs_main_Probe_Filters_0_SecurityMask

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_emacs_main_Probe	0xFFD14400	0xFFD1445C

Offset: 0x5C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													FILTERS_0_SECURITY-MASK RW 0x0		

Probe_emacs_main_Probe_Filters_0_SecurityMask Fields

Bit	Name	Description	Access	Reset
2:0	FILTERS_0_SECURITYMASK	Register SecurityMask is contains the security mask used to filter packets. A packet is a candidate when: packet.Security & Security-Mask = SecurityBase & SecurityMasks.	RW	0x0

Probe_emacs_main_Probe_Filters_0_Opcode

Packet Probe register Opcode is a 4-bit register that selects candidate packets based on packet opcodes (0 disables the filter):

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_emacs_main_Probe	0xFFD14400	0xFFD14460

Offset: 0x60

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												URGEN RW 0x0	LOCKEN RW 0x0	WREN RW 0x0	RDEN RW 0x0

Probe_emacs_main_Probe_Filters_0_Opcode Fields

Bit	Name	Description	Access	Reset
3	URGEN	Selects URG packets (urgency).	RW	0x0

Bit	Name	Description	Access	Reset
2	LOCKEN	Selects RDX-WR, RDL, WRC and Linked sequence.	RW	0x0
1	WREN	Selects WR packets.	RW	0x0
0	RDEN	Selects RD packets.	RW	0x0

Probe_emacs_main_Probe_Filters_0_Status

Register Status is 2-bit register that selects candidate packets based on packet status.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_emacs_main_Probe	0xFFD14400	0xFFD14464

Offset: 0x64

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														RSPEN RW 0x0	REQEN RW 0x0

Probe_emacs_main_Probe_Filters_0_Status Fields

Bit	Name	Description	Access	Reset
1	RSPEN	Selects RSP and FAIL-CONT status packets.	RW	0x0
0	REQEN	Selects REQ status packets.	RW	0x0

Probe_emacs_main_Probe_Filters_0_Length

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_emacs_main_Probe	0xFFD14400	0xFFD14468

Offset: 0x68

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												FILTERS_0_LENGTH RW 0x0			

Probe_emacs_main_Probe_Filters_0_Length Fields

Bit	Name	Description	Access	Reset
3:0	FILTERS_0_LENGTH	Register Length is 4-bit register that selects candidate packets if their number of bytes is less than or equal to 2**Length.	RW	0x0

Probe_emacs_main_Probe_Filters_0_Urgency

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_emacs_main_Probe	0xFFD14400	0xFFD1446C

Offset: 0x6C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														FILTERS_0_URGENCY RW 0x0	

Probe_emacs_main_Probe_Filters_0_Urgency Fields

Bit	Name	Description	Access	Reset
1:0	FILTERS_0_URGENCY	Register Urgency contains the minimum urgency level used to filter packets. A packet is a candidate when its socket urgency is greater than or equal to the urgency specified in the register.	RW	0x0

Probe_emacs_main_Probe_Filters_1_RouteIdBase

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_emacs_main_Probe	0xFFD14400	0xFFD14480

Offset: 0x80

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													FILTERS_1_ROUTEIDBASE RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FILTERS_1_ROUTEIDBASE RW 0x0															

Probe_emacs_main_Probe_Filters_1_RouteIdBase Fields

Bit	Name	Description	Access	Reset
18:0	FILTERS_1_ROUTEIDBASE	Register RouteIdBase contains the RouteId-lsbFilterRouteId bits base used to filter packets.	RW	0x0

Probe_emacs_main_Probe_Filters_1_RouteIdMask

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_emacs_main_Probe	0xFFD14400	0xFFD14484

Offset: 0x84

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													FILTERS_1_ROUTEIDMASK RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FILTERS_1_ROUTEIDMASK RW 0x0															

Probe_emacs_main_Probe_Filters_1_RouteIdMask Fields

Bit	Name	Description	Access	Reset
18:0	FILTERS_1_ROUTEIDMASK	Register RouteIdMask contains the RouteId-lsbFilterRouteId mask used to filter packets. A packet is a candidate when packet.RouteId>> lsbFilterRouteId & RouteIdMask = RouteIdBase & RouteIdMask.	RW	0x0

Probe_emacs_main_Probe_Filters_1_AddrBase_Low

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_emacs_main_Probe	0xFFD14400	0xFFD14488

Offset: 0x88

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FILTERS_1_ADDRBASE_LOW RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FILTERS_1_ADDRBASE_LOW RW 0x0															

Probe_emacs_main_Probe_Filters_1_AddrBase_Low Fields

Bit	Name	Description	Access	Reset
31:0	FILTERS_1_ADDRBASE_LOW	Address LSB register.	RW	0x0

Probe_emacs_main_Probe_Filters_1_WindowSize

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_emacs_main_Probe	0xFFD14400	0xFFD14490

Offset: 0x90

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										FILTERS_1_WINDOWSIZE RW 0x0					

Probe_emacs_main_Probe_Filters_1_WindowSize Fields

Bit	Name	Description	Access	Reset
5:0	FILTERS_1_WINDOWSIZE	Register WindowSize contains the encoded address mask used to filter packets. The effective Mask value is equal to ~(2max(WindowSize, packet.Len) - 1). A packet is a candidate when packet.Addr & Mask = AddrBase & Mask. This allows filteringof packets having an intersection with the AddrBase/ WindowSize burst aligned region, even if the region is smaller than the packet.	RW	0x0

Probe_emacs_main_Probe_Filters_1_SecurityBase

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_emacs_main_Probe	0xFFD14400	0xFFD14494

Offset: 0x94

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													FILTERS_1_SECURITY-BASE RW 0x0		

Probe_emacs_main_Probe_Filters_1_SecurityBase Fields

Bit	Name	Description	Access	Reset
2:0	FILTERS_1_SECURITYBASE	Register SecurityBase contains the security base used to filter packets.	RW	0x0

Probe_emacs_main_Probe_Filters_1_SecurityMask

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_emacs_main_Probe	0xFFD14400	0xFFD14498

Offset: 0x98

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													FILTERS_1_SECURITY-MASK RW 0x0		

Probe_emacs_main_Probe_Filters_1_SecurityMask Fields

Bit	Name	Description	Access	Reset
2:0	FILTERS_1_SECURITYMASK	Register SecurityMask is contains the security mask used to filter packets. A packet is a candidate when: packet.Security & Security-Mask = SecurityBase & SecurityMasks.	RW	0x0

Probe_emacs_main_Probe_Filters_1_Opcode

Packet Probe register Opcode is a 4-bit register that selects candidate packets based on packet opcodes (0 disables the filter):

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_emacs_main_Probe	0xFFD14400	0xFFD1449C

Offset: 0x9C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												URGEN RW 0x0	LOCKEN RW 0x0	WREN RW 0x0	RDEN RW 0x0

Probe_emacs_main_Probe_Filters_1_Opcode Fields

Bit	Name	Description	Access	Reset
3	URGEN	Selects URG packets (urgency).	RW	0x0
2	LOCKEN	Selects RDX-WR, RDL, WRC and Linked sequence.	RW	0x0
1	WREN	Selects WR packets.	RW	0x0
0	RDEN	Selects RD packets.	RW	0x0

Probe_emacs_main_Probe_Filters_1_Status

Register Status is 2-bit register that selects candidate packets based on packet status.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_emacs_main_Probe	0xFFD14400	0xFFD144A0

Offset: 0xA0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														RSPEN RW 0x0	REQEN RW 0x0

Probe_emacs_main_Probe_Filters_1_Status Fields

Bit	Name	Description	Access	Reset
1	RSPEN	Selects RSP and FAIL-CONT status packets.	RW	0x0
0	REQEN	Selects REQ status packets.	RW	0x0

Probe_emacs_main_Probe_Filters_1_Length

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_emacs_main_Probe	0xFFD14400	0xFFD144A4

Offset: 0xA4

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												FILTERS_1_LENGTH RW 0x0			

Probe_emacs_main_Probe_Filters_1_Length Fields

Bit	Name	Description	Access	Reset
3:0	FILTERS_1_LENGTH	Register Length is 4-bit register that selects candidate packets if their number of bytes is less than or equal to 2**Length.	RW	0x0

Probe_emacs_main_Probe_Filters_1_Urgency

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_emacs_main_Probe	0xFFD14400	0xFFD144A8

Offset: 0xA8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														FILTERS_1_URGENCY RW 0x0	

Probe_emacs_main_Probe_Filters_1_Urgency Fields

Bit	Name	Description	Access	Reset
1:0	FILTERS_1_URGENCY	Register Urgency contains the minimum urgency level used to filter packets. A packet is a candidate when its socket urgency is greater than or equal to the urgency specified in the register.	RW	0x0

Probe_emacs_main_Probe_Counters_0_PortSel

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_emacs_main_Probe	0xFFD14400	0xFFD14534

Offset: 0x134

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															COUNTERS_0_PORTSEL RW 0x0

Probe_emacs_main_Probe_Counters_0_PortSel Fields

Bit	Name	Description	Access	Reset
0	COUNTERS_0_PORTSEL	Register PortSel indicates which NTTP link is associated with the counter. The register can be changed at any time, with the change effective immediately. The LUT and FILTx sources do not depend on this NTTP port selection.	RW	0x0

Probe_emacs_main_Probe_Counters_0_Src

Register CntSrc indicates the event source used to increment the counter. Unassigned values (non-existing Press level or ExtEvent index, or unimplemented Filter) are equivalent to OFF.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_emacs_main_Probe	0xFFD14400	0xFFD14538

Offset: 0x138

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										EXTEVENT RW 0x0	INTEVENT RW 0x0				

Probe_emacs_main_Probe_Counters_0_Src Fields

Bit	Name	Description	Access	Reset
5	EXTEVENT	When set to 1 counts the cycles where ExtEvent[IntEvent] = 1. It exists when nExtEvent > 0.	RW	0x0
4:0	INTEVENT	Internal packet event	RW	0x0

Probe_emacs_main_Probe_Counters_0_AlarmMode

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_emacs_main_Probe	0xFFD14400	0xFFD1453C

Offset: 0x13C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														COUNTERS_0_ALARMMODE RW 0x0	

Probe_emacs_main_Probe_Counters_0_AlarmMode Fields

Bit	Name	Description	Access	Reset
1:0	COUNTERS_0_ALARMMODE	Register AlarmMode is a 2-bit register that is present when parameter statisticsCounterAlarm is set to True. The register defines the statistics-alarm behavior of the counter.	RW	0x0

Probe_emacs_main_Probe_Counters_0_Val

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_emacs_main_Probe	0xFFD14400	0xFFD14540

Offset: 0x140

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COUNTERS_0_VAL RO 0x0															

Probe_emacs_main_Probe_Counters_0_Val Fields

Bit	Name	Description	Access	Reset
15:0	COUNTERS_0_VAL	Register Val is a read-only register that is always present. The register containsthe statistics counter value either pending StatAlarm output, or when statisticscollection is suspended subsequent to triggers or signal statSuspend.	RO	0x0

Probe_emacs_main_Probe_Counters_1_PortSel

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_emacs_main_Probe	0xFFD14400	0xFFD14548

Offset: 0x148

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															COUNTERS_1_PORTSEL RW 0x0

Probe_emacs_main_Probe_Counters_1_PortSel Fields

Bit	Name	Description	Access	Reset
0	COUNTERS_1_PORTSEL	Register PortSel indicates which NTTP link is associated with the counter. The register can be changed at any time, with the change effective immediately. The LUT and FILTx sources do not depend on this NTTP port selection.	RW	0x0

Probe_emacs_main_Probe_Counters_1_Src

Register CntSrc indicates the event source used to increment the counter. Unassigned values (non-existing Press level or ExtEvent index, or unimplemented Filter) are equivalent to OFF.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_emacs_main_Probe	0xFFD14400	0xFFD1454C

Offset: 0x14C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										EXTEVENT RW 0x0	INTEVENT RW 0x0				

Probe_emacs_main_Probe_Counters_1_Src Fields

Bit	Name	Description	Access	Reset
5	EXTEVENT	When set to 1 counts the cycles where ExtEvent[IntEvent] = 1. It exists when nExtEvent > 0.	RW	0x0
4:0	INTEVENT	Internal packet event	RW	0x0

Probe_emacs_main_Probe_Counters_1_AlarmMode

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_emacs_main_Probe	0xFFD14400	0xFFD14550

Offset: 0x150

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														COUNTERS_1_ALARMMODE RW 0x0	

Probe_emacs_main_Probe_Counters_1_AlarmMode Fields

Bit	Name	Description	Access	Reset
1:0	COUNTERS_1_ALARMMODE	Register AlarmMode is a 2-bit register that is present when parameter statisticsCounterAlarm is set to True. The register defines the statistics-alarm behavior of the counter.	RW	0x0

Probe_emacs_main_Probe_Counters_1_Val

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_emacs_main_Probe	0xFFD14400	0xFFD14554

Offset: 0x154

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COUNTERS_1_VAL RO 0x0															

Probe_emacs_main_Probe_Counters_1_Val Fields

Bit	Name	Description	Access	Reset
15:0	COUNTERS_1_VAL	Register Val is a read-only register that is always present. The register containsthe statistics counter value either pending StatAlarm output, or when statisticscollection is suspended subsequent to triggers or signal statSuspend.	RO	0x0

Probe_emacs_main_Probe_Counters_2_PortSel

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_emacs_main_Probe	0xFFD14400	0xFFD1455C

Offset: 0x15C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															COUNTERS_2_PORTSEL RW 0x0

Probe_emacs_main_Probe_Counters_2_PortSel Fields

Bit	Name	Description	Access	Reset
0	COUNTERS_2_PORTSEL	Register PortSel indicates which NTTP link is associated with the counter. The register can be changed at any time, with the change effective immediately. The LUT and FILTx sources do not depend on this NTTP port selection.	RW	0x0

Probe_emacs_main_Probe_Counters_2_Src

Register CntSrc indicates the event source used to increment the counter. Unassigned values (non-existing Press level or ExtEvent index, or unimplemented Filter) are equivalent to OFF.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_emacs_main_Probe	0xFFD14400	0xFFD14560

Offset: 0x160

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										EXTEVENT RW 0x0	INTEVENT RW 0x0				

Probe_emacs_main_Probe_Counters_2_Src Fields

Bit	Name	Description	Access	Reset
5	EXTEVENT	When set to 1 counts the cycles where ExtEvent[IntEvent] = 1. It exists when nExtEvent > 0.	RW	0x0
4:0	INTEVENT	Internal packet event	RW	0x0

Probe_emacs_main_Probe_Counters_2_AlarmMode

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_emacs_main_Probe	0xFFD14400	0xFFD14564

Offset: 0x164

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														COUNTERS_2_ALARMMODE RW 0x0	

Probe_emacs_main_Probe_Counters_2_AlarmMode Fields

Bit	Name	Description	Access	Reset
1:0	COUNTERS_2_ALARMMODE	Register AlarmMode is a 2-bit register that is present when parameter statisticsCounterAlarm is set to True. The register defines the statistics-alarm behavior of the counter.	RW	0x0

Probe_emacs_main_Probe_Counters_2_Val

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_emacs_main_Probe	0xFFD14400	0xFFD14568

Offset: 0x168

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COUNTERS_2_VAL RO 0x0															

Probe_emacs_main_Probe_Counters_2_Val Fields

Bit	Name	Description	Access	Reset
15:0	COUNTERS_2_VAL	Register Val is a read-only register that is always present. The register containsthe statistics counter value either pending StatAlarm output, or when statisticscollection is suspended subsequent to triggers or signal statSuspend.	RO	0x0

Probe_emacs_main_Probe_Counters_3_PortSel

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_emacs_main_Probe	0xFFD14400	0xFFD14570

Offset: 0x170

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															COUNTERS_3_PORTSEL RW 0x0

Probe_emacs_main_Probe_Counters_3_PortSel Fields

Bit	Name	Description	Access	Reset
0	COUNTERS_3_PORTSEL	Register PortSel indicates which NTTP link is associated with the counter. The register can be changed at any time, with the change effective immediately. The LUT and FILTx sources do not depend on this NTTP port selection.	RW	0x0

Probe_emacs_main_Probe_Counters_3_Src

Register CntSrc indicates the event source used to increment the counter. Unassigned values (non-existing Press level or ExtEvent index, or unimplemented Filter) are equivalent to OFF.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_emacs_main_Probe	0xFFD14400	0xFFD14574

Offset: 0x174

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										EXTEVENT RW 0x0	INTEVENT RW 0x0				

Probe_emacs_main_Probe_Counters_3_Src Fields

Bit	Name	Description	Access	Reset
5	EXTEVENT	When set to 1 counts the cycles where ExtEvent[IntEvent] = 1. It exists when nExtEvent > 0.	RW	0x0
4:0	INTEVENT	Internal packet event	RW	0x0

Probe_emacs_main_Probe_Counters_3_AlarmMode

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_emacs_main_Probe	0xFFD14400	0xFFD14578

Offset: 0x178

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														COUNTERS_3_ALARMMODE RW 0x0	

Probe_emacs_main_Probe_Counters_3_AlarmMode Fields

Bit	Name	Description	Access	Reset
1:0	COUNTERS_3_ALARMMODE	Register AlarmMode is a 2-bit register that is present when parameter statisticsCounterAlarm is set to True. The register defines the statistics-alarm behavior of the counter.	RW	0x0

Probe_emacs_main_Probe_Counters_3_Val

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_emacs_main_Probe	0xFFD14400	0xFFD1457C

Offset: 0x17C

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COUNTERS_3_VAL RO 0x0															

Probe_emacs_main_Probe_Counters_3_Val Fields

Bit	Name	Description	Access	Reset
15:0	COUNTERS_3_VAL	Register Val is a read-only register that is always present. The register containsthe statistics counter value either pending StatAlarm output, or when statisticscollection is suspended subsequent to triggers or signal statSuspend.	RO	0x0

noc_mpu_m0_Probe_emacs_main_TransactionStatProfiler Address Map

Module Instance	Base Address	End Address
i_noc_mpu_m0_Probe_emacs_main_TransactionStatProfiler	0xFFD14800	0xFFD148FF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
Probe_emacs_main_TransactionStatProfiler_Id_CoreId on page 7-594	0x0	32	RO	0xA6B7960A	
Probe_emacs_main_TransactionStatProfiler_Id_RevisionId on page 7-595	0x4	32	RO	0x129FF00	
Probe_emacs_main_TransactionStatProfiler_En on page 7-595	0x8	32	RW	0x0	
Probe_emacs_main_TransactionStatProfiler_Mode on page 7-596	0xC	32	RW	0x0	
Probe_emacs_main_TransactionStatProfiler_Thresholds_0_0 on page 7-597	0x2C	32	RW	0x0	
Probe_emacs_main_TransactionStatProfiler_Thresholds_0_1 on page 7-598	0x30	32	RW	0x0	
Probe_emacs_main_TransactionStatProfiler_Thresholds_0_2 on page 7-598	0x34	32	RW	0x0	
Probe_emacs_main_TransactionStatProfiler_OverflowStatus on page 7-599	0x6C	32	RO	0x0	
Probe_emacs_main_TransactionStatProfiler_OverflowReset on page 7-600	0x70	32	RW	0x0	

Register	Offset	Width	Access	Reset Value	Description
Probe_emacs_main_TransactionStatProfiler_PendingEventMode on page 7-601	0x74	32	RW	0x0	
Probe_emacs_main_TransactionStatProfiler_PreScaler on page 7-602	0x78	32	RW	0x0	

noc_mpu_m0_Probe_emacs_main_TransactionStatProfiler Summary

Base Address: 0xFFD14800

Register Address Offset	Bit Fields															
i_noc_mpu_m0_Probe_emacs_main_TransactionStatProfiler																
Probe_emacs_main_TransactionStatProfiler_Id_CoreId 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	CORECHECKSUM RO 0xA6B796															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CORECHECKSUM RO 0xA6B796								CORETYPEID RO 0xA							
Probe_emacs_main_TransactionStatProfiler_Id_RevisionId 0x4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	FLEXNOCID RO 0x129FF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FLEXNOCID RO 0x129FF								USERID RO 0x0							
Probe_emacs_main_TransactionStatProfiler_En 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															EN RW 0x0

Register Address Offset	Bit Fields															
Probe_emacs_main_TransactionStat-Profiler_Mode 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															MODE RW 0x0
Probe_emacs_main_TransactionStat-Profiler_Thresholds_0_0 0x2C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														THRESHOLDS_0_0 RW 0x0	
Probe_emacs_main_TransactionStat-Profiler_Thresholds_0_1 0x30	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														THRESHOLDS_0_1 RW 0x0	
Probe_emacs_main_TransactionStat-Profiler_Thresholds_0_2 0x34	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														THRESHOLDS_0_2 RW 0x0	
Probe_emacs_main_TransactionStat-Profiler_OverflowStatus 0x6C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															OVERFLOW STATUS RO 0x0
Probe_emacs_main_TransactionStat-Profiler_OverflowReset 0x70	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															OVERFLOW RESET RW 0x0

Register Address Offset	Bit Fields															
Probe_emacs_main_TransactionStat-Profiler_PendingEvent-Mode 0x74	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															PENDINGEVENTMODE RW 0x0
Probe_emacs_main_TransactionStat-Profiler_PreScaler 0x78	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								PRESCALER RW 0x0							

Probe_emacs_main_TransactionStatProfiler_Id_CoreId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_emacs_main_TransactionStatProfiler	0xFFD14800	0xFFD14800

Offset: 0x0

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CORECHECKSUM RO 0xA6B796															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CORECHECKSUM RO 0xA6B796								CORETYPEID RO 0xA							

Probe_emacs_main_TransactionStatProfiler_Id_CoreId Fields

Bit	Name	Description	Access	Reset
31:8	CORECHECKSUM	Field containing a checksum of the parameters of the IP.	RO	0xA6B796

Bit	Name	Description	Access	Reset
7:0	CORETYPEID	Field identifying the type of IP.	RO	0xA

Probe_emacs_main_TransactionStatProfiler_Id_RevisionId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_emacs_main_TransactionStatProfiler	0xFFD14800	0xFFD14804

Offset: 0x4

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLEXNOCID RO 0x129FF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLEXNOCID RO 0x129FF								USERID RO 0x0							

Probe_emacs_main_TransactionStatProfiler_Id_RevisionId Fields

Bit	Name	Description	Access	Reset
31:8	FLEXNOCID	Field containing the build revision of the software used to generate the IP HDL code.	RO	0x129FF
7:0	USERID	Field containing a user defined value, not used anywhere inside the IP itself.	RO	0x0

Probe_emacs_main_TransactionStatProfiler_En

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_emacs_main_TransactionStatProfiler	0xFFD14800	0xFFD14808

Offset: 0x8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															EN RW 0x0

Probe_emacs_main_TransactionStatProfiler_En Fields

Bit	Name	Description	Access	Reset
0	EN	Register En is a 1-bit register that enables the transaction probe counter unit.	RW	0x0

Probe_emacs_main_TransactionStatProfiler_Mode

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_emacs_main_TransactionStatProfiler	0xFFD14800	0xFFD1480C

Offset: 0xC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															MODE RW 0x0

Probe_emacs_main_TransactionStatProfiler_Mode Fields

Bit	Name	Description	Access	Reset
0	MODE	Register Mode sets the counting mode per observed port. Each bit per observation port defines the incrementing mode. (Mode = 0 for Delay, Mode = 1 for Pending)	RW	0x0

Probe_emacs_main_TransactionStatProfiler_Thresholds_0_0

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_emacs_main_TransactionStatProfiler	0xFFD14800	0xFFD1482C

Offset: 0x2C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															THRESHOLDS_0_0 RW 0x0

Probe_emacs_main_TransactionStatProfiler_Thresholds_0_0 Fields

Bit	Name	Description	Access	Reset
1:0	THRESHOLDS_0_0	Register Thresholds_i_j contains the threshold index "0" that allows computation of threshold values.	RW	0x0

Probe_emacs_main_TransactionStatProfiler_Thresholds_0_1

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_emacs_main_TransactionStatProfiler	0xFFD14800	0xFFD14830

Offset: 0x30

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														THRESHOLDS_0_1 RW 0x0	

Probe_emacs_main_TransactionStatProfiler_Thresholds_0_1 Fields

Bit	Name	Description	Access	Reset
1:0	THRESHOLDS_0_1	Register Thresholds_i_j contains the threshold index "1" that allows computation of threshold values.	RW	0x0

Probe_emacs_main_TransactionStatProfiler_Thresholds_0_2

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_emacs_main_TransactionStatProfiler	0xFFD14800	0xFFD14834

Offset: 0x34

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														THRESHOLDS_0_2 RW 0x0	

Probe_emacs_main_TransactionStatProfiler_Thresholds_0_2 Fields

Bit	Name	Description	Access	Reset
1:0	THRESHOLDS_0_2	Register Thresholds_i_j contains the threshold index "2" that allows computation of threshold values.	RW	0x0

Probe_emacs_main_TransactionStatProfiler_OverflowStatus

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_emacs_main_TransactionStatProfiler	0xFFD14800	0xFFD1486C

Offset: 0x6C

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															OVERFLOW STATUS RO 0x0

Probe_emacs_main_TransactionStatProfiler_OverflowStatus Fields

Bit	Name	Description	Access	Reset
0	OVERFLOWSTATUS	Bit n of register OverflowStatus is set to 1 if a start event occurs on observed port n and eitherof the following conditions occurs: All tenure counters allocated to the port are already in use. No tenure lines have been allocated to the port. The number of bits in this register is equal to the value set for parameter nObservable.	RO	0x0

Probe_emacs_main_TransactionStatProfiler_OverflowReset

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_emacs_main_TransactionStatProfiler	0xFFD14800	0xFFD14870

Offset: 0x70

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															OVERFLOW RESET RW 0x0

Probe_emacs_main_TransactionStatProfiler_OverflowReset Fields

Bit	Name	Description	Access	Reset
0	OVERFLOWRESET	Register OverflowReset is a pulse register that clears overflow status bits per observed port on each write access. OverflowReset = nObservable. Writing 0x2 clears the overflow status of observed port 1.	RW	0x0

Probe_emacs_main_TransactionStatProfiler_PendingEventMode

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_emacs_main_TransactionStatProfiler	0xFFD14800	0xFFD14874

Offset: 0x74

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															PENDINGEVENTMODE RW 0x0

Probe_emacs_main_TransactionStatProfiler_PendingEventMode Fields

Bit	Name	Description	Access	Reset
0	PENDINGEVENTMODE	Register pendingEventMode is a 1-bit register that configures the pending event mode. When set to 0 (CYCLE), and when register mode is set to PENDING, the pending event is generated on each cycle when the counter is greater than zero.When set to 1 (STOP) the pending event is generated on each stop event.	RW	0x0

Probe_emacs_main_TransactionStatProfiler_PreScaler

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_emacs_main_TransactionStatProfiler	0xFFD14800	0xFFD14878

Offset: 0x78

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								PRESCALER RW 0x0							

Probe_emacs_main_TransactionStatProfiler_PreScaler Fields

Bit	Name	Description	Access	Reset
7:0	PRESCALER	8Register Prescaler is an-bit pre-scaling register that accepts any pre-scaling value between 1 (default) and 256. If set to 0, pre-scaling is disabled. If set to any other supported value "n", the threshold counter value is divided by (n + 1).	RW	0x0

noc_mpu_m0_cs_obs_at_main_AtbEndPoint Address Map

Module Instance	Base Address	End Address
i_noc_mpu_m0_cs_obs_at_main_AtbEndPoint	0xFFD14900	0xFFD1497F

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
cs_obs_at_main_AtbEndPoint_Id_CoreId on page 7-605	0x0	32	RO	0xD46B8707	
cs_obs_at_main_AtbEndPoint_Id_RevisionId on page 7-605	0x4	32	RO	0x129FF00	
cs_obs_at_main_AtbEndPoint_AtbId on page 7-606	0x8	32	RW	0x0	
cs_obs_at_main_AtbEndPoint_AtbEn on page 7-607	0xC	32	RW	0x0	
cs_obs_at_main_AtbEndPoint_SyncPeriod on page 7-607	0x10	32	RW	0x0	

noc_mpu_m0_cs_obs_at_main_AtbEndPoint Summary

Base Address: 0xFFD14900

Register Address Offset	Bit Fields															
i_noc_mpu_m0_cs_obs_at_main_AtbEndPoint																
cs_obs_at_main_AtbEndPoint_Id_CoreId 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	CORECHECKSUM RO 0xD46B87															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CORECHECKSUM RO 0xD46B87								CORETYPEID RO 0x7							
cs_obs_at_main_AtbEndPoint_Id_RevisionId 0x4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	FLEXNOCID RO 0x129FF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FLEXNOCID RO 0x129FF								USERID RO 0x0							
cs_obs_at_main_AtbEndPoint_AtbId 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved									ATBID RW 0x0						
cs_obs_at_main_AtbEndPoint_AtbEn 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															ATBEN RW 0x0
cs_obs_at_main_AtbEndPoint_SyncPeriod 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved											SYNCPERIOD RW 0x0				

cs_obs_at_main_AtbEndPoint_Id_CoreId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_cs_obs_at_main_AtbEndPoint	0xFFD14900	0xFFD14900

Offset: 0x0

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CORECHECKSUM RO 0xD46B87															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CORECHECKSUM RO 0xD46B87								CORETYPEID RO 0x7							

cs_obs_at_main_AtbEndPoint_Id_CoreId Fields

Bit	Name	Description	Access	Reset
31:8	CORECHECKSUM	Field containing a checksum of the parameters of the IP.	RO	0xD46B87
7:0	CORETYPEID	Field identifying the type of IP.	RO	0x7

cs_obs_at_main_AtbEndPoint_Id_RevisionId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_cs_obs_at_main_AtbEndPoint	0xFFD14900	0xFFD14904

Offset: 0x4

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLEXNOCID RO 0x129FF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLEXNOCID RO 0x129FF								USERID RO 0x0							

cs_obs_at_main_AtbEndPoint_Id_RevisionId Fields

Bit	Name	Description	Access	Reset
31:8	FLEXNOCID	Field containing the build revision of the software used to generate the IP HDL code.	RO	0x129FF
7:0	USERID	Field containing a user defined value, not used anywhere inside the IP itself.	RO	0x0

cs_obs_at_main_AtbEndPoint_AtbId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_cs_obs_at_main_AtbEndPoint	0xFFD14900	0xFFD14908

Offset: 0x8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									ATBID RW 0x0						

cs_obs_at_main_AtbEndPoint_AtbId Fields

Bit	Name	Description	Access	Reset
6:0	ATBID	ATB AtId	RW	0x0

cs_obs_at_main_AtbEndPoint_AtbEn

Module Instance	Base Address	Register Address
i_noc_mpu_m0_cs_obs_at_main_AtbEndPoint	0xFFD14900	0xFFD1490C

Offset: 0xC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															ATBEN RW 0x0

cs_obs_at_main_AtbEndPoint_AtbEn Fields

Bit	Name	Description	Access	Reset
0	ATBEN	ATB Unit Enable	RW	0x0

cs_obs_at_main_AtbEndPoint_SyncPeriod

Module Instance	Base Address	Register Address
i_noc_mpu_m0_cs_obs_at_main_AtbEndPoint	0xFFD14900	0xFFD14910

Offset: 0x10

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved											SYNCPERIOD RW 0x0				

cs_obs_at_main_AtbEndPoint_SyncPeriod Fields

Bit	Name	Description	Access	Reset
4:0	SYNCPERIOD	ATB Synchro Period	RW	0x0

noc_mpu_m0_cs_obs_at_main_ErrorLogger_0 Address Map

Module Instance	Base Address	End Address
i_noc_mpu_m0_cs_obs_at_main_ErrorLogger_0	0xFFD14980	0xFFD14FFF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
cs_obs_at_main_ErrorLogger_0_Id_CoreId on page 7-611	0x0	32	RO	0x91311E0D	

Register	Offset	Width	Access	Reset Value	Description
cs_obs_at_main_ErrorLogger_0_Id_RevisionId on page 7-612	0x4	32	RO	0x129FF00	
cs_obs_at_main_ErrorLogger_0_FaultEn on page 7-613	0x8	32	RW	0x0	
cs_obs_at_main_ErrorLogger_0_ErrVld on page 7-614	0xC	32	RO	0x0	
cs_obs_at_main_ErrorLogger_0_ErrClr on page 7-614	0x10	32	RW	0x0	
cs_obs_at_main_ErrorLogger_0_ErrLog0 on page 7-615	0x14	32	RO	0x80000000	Stores NTTP packet header fields Lock, Opc, ErrCode, Len1 and indicates version of NTTP transport protocol
cs_obs_at_main_ErrorLogger_0_ErrLog1 on page 7-616	0x18	32	RO	0x0	
cs_obs_at_main_ErrorLogger_0_ErrLog3 on page 7-617	0x20	32	RO	0x0	
cs_obs_at_main_ErrorLogger_0_ErrLog5 on page 7-618	0x28	32	RO	0x0	
cs_obs_at_main_ErrorLogger_0_ErrLog7 on page 7-618	0x30	32	RO	0x0	

noc_mpu_m0_cs_obs_at_main_ErrorLogger_0 Summary

Base Address: 0xFFD14980

Register Address Offset	Bit Fields															
i_noc_mpu_m0_cs_obs_at_main_ErrorLogger_0																
cs_obs_at_main_ErrorLogger_0_Id_CoreId 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	CORECHECKSUM RO 0x91311E															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CORECHECKSUM RO 0x91311E								CORETYPEID RO 0xD							
cs_obs_at_main_ErrorLogger_0_Id_RevisionId 0x4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	FLEXNOCID RO 0x129FF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FLEXNOCID RO 0x129FF								USERID RO 0x0							
cs_obs_at_main_ErrorLogger_0_FaultEn 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															FAULTEN RW 0x0
cs_obs_at_main_ErrorLogger_0_ErrVld 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															ERRVLD RO 0x0
cs_obs_at_main_ErrorLogger_0_ErrClr 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															ERRCLR RW 0x0

Register Address Offset	Bit Fields															
cs_obs_at_main_ErrorLogger_0_ErrLog0 0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	FORMAT RO 0x1	Reserved							LEN1 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved					ERRCODE RO 0x0			Reserved			OPC RO 0x0				LOCK RO 0x0
cs_obs_at_main_ErrorLogger_0_ErrLog1 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved													ERRLOG1 RO 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ERRLOG1 RO 0x0															
cs_obs_at_main_ErrorLogger_0_ErrLog3 0x20	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ERRLOG3 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ERRLOG3 RO 0x0															
cs_obs_at_main_ErrorLogger_0_ErrLog5 0x28	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved					ERRLOG5 RO 0x0										
cs_obs_at_main_ErrorLogger_0_ErrLog7 0x30	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													ERRLOG7 RO 0x0		

cs_obs_at_main_ErrorLogger_0_Id_CoreId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_cs_obs_at_main_ErrorLogger_0	0xFFD14980	0xFFD14980

Offset: 0x0

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CORECHECKSUM RO 0x91311E															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CORECHECKSUM RO 0x91311E								CORETYPEID RO 0xD							

cs_obs_at_main_ErrorLogger_0_Id_CoreId Fields

Bit	Name	Description	Access	Reset
31:8	CORECHECKSUM	Field containing a checksum of the parameters of the IP.	RO	0x91311E
7:0	CORETYPEID	Field identifying the type of IP.	RO	0xD

cs_obs_at_main_ErrorLogger_0_Id_RevisionId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_cs_obs_at_main_ErrorLogger_0	0xFFD14980	0xFFD14984

Offset: 0x4

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLEXNOCID RO 0x129FF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLEXNOCID RO 0x129FF								USERID RO 0x0							

cs_obs_at_main_ErrorLogger_0_Id_RevisionId Fields

Bit	Name	Description	Access	Reset
31:8	FLEXNOCID	Field containing the build revision of the software used to generate the IP HDL code.	RO	0x129FF
7:0	USERID	Field containing a user defined value, not used anywhere inside the IP itself.	RO	0x0

cs_obs_at_main_ErrorLogger_0_FaultEn

Module Instance	Base Address	Register Address
i_noc_mpu_m0_cs_obs_at_main_ErrorLogger_0	0xFFD14980	0xFFD14988

Offset: 0x8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															FAULTEN RW 0x0

cs_obs_at_main_ErrorLogger_0_FaultEn Fields

Bit	Name	Description	Access	Reset
0	FAULTEN	Set to 1 to enable output signal Fault. Fault asserted when ErrVld is 1.	RW	0x0

cs_obs_at_main_ErrorLogger_0_ErrVld

Module Instance	Base Address	Register Address
i_noc_mpu_m0_cs_obs_at_main_ErrorLogger_0	0xFFD14980	0xFFD1498C

Offset: 0xC

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															ERRVLD RO 0x0

cs_obs_at_main_ErrorLogger_0_ErrVld Fields

Bit	Name	Description	Access	Reset
0	ERRVLD	1 indicates an error has been logged	RO	0x0

cs_obs_at_main_ErrorLogger_0_ErrClr

Module Instance	Base Address	Register Address
i_noc_mpu_m0_cs_obs_at_main_ErrorLogger_0	0xFFD14980	0xFFD14990

Offset: 0x10

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															ERRCLR RW 0x0

cs_obs_at_main_ErrorLogger_0_ErrClr Fields

Bit	Name	Description	Access	Reset
0	ERRCLR	Set to 1 to clear ErrVld. NOTE The written value is not stored in ErrVld. A read always returns 0.	RW	0x0

cs_obs_at_main_ErrorLogger_0_ErrLog0

Stores NTTP packet header fields Lock, Opc, ErrCode, Len1 and indicates version of NTTP transport protocol

Module Instance	Base Address	Register Address
i_noc_mpu_m0_cs_obs_at_main_ErrorLogger_0	0xFFD14980	0xFFD14994

Offset: 0x14

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FORMAT RO 0x1	Reserved							LEN1 RO 0x0							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					ERRCODE RO 0x0			Reserved			OPC RO 0x0				LOCK RO 0x0

cs_obs_at_main_ErrorLogger_0_ErrLog0 Fields

Bit	Name	Description	Access	Reset
31	FORMAT	NTTP transport protocol version	RO	0x1
23:16	LEN1	Len1	RO	0x0
10:8	ERRCODE	ErrCode	RO	0x0
4:1	OPC	Opc	RO	0x0
0	LOCK	Lock	RO	0x0

cs_obs_at_main_ErrorLogger_0_ErrLog1

Module Instance	Base Address	Register Address
i_noc_mpu_m0_cs_obs_at_main_ErrorLogger_0	0xFFD14980	0xFFD14998

Offset: 0x18

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													ERRLOG1 RO 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ERRLOG1 RO 0x0															

cs_obs_at_main_ErrorLogger_0_ErrLog1 Fields

Bit	Name	Description	Access	Reset
18:0	ERRLOG1	Stores NTTP packet header field RouteId (LSBs) of the logged error	RO	0x0

cs_obs_at_main_ErrorLogger_0_ErrLog3

Module Instance	Base Address	Register Address
i_noc_mpu_m0_cs_obs_at_main_ErrorLogger_0	0xFFD14980	0xFFD149A0

Offset: 0x20

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ERRLOG3 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ERRLOG3 RO 0x0															

cs_obs_at_main_ErrorLogger_0_ErrLog3 Fields

Bit	Name	Description	Access	Reset
31:0	ERRLOG3	Stores NTTP packet header field Addr (LSBs) of the logged error	RO	0x0

cs_obs_at_main_ErrorLogger_0_ErrLog5

Module Instance	Base Address	Register Address
i_noc_mpu_m0_cs_obs_at_main_ErrorLogger_0	0xFFD14980	0xFFD149A8

Offset: 0x28

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					ERRLOG5 RO 0x0										

cs_obs_at_main_ErrorLogger_0_ErrLog5 Fields

Bit	Name	Description	Access	Reset
10:0	ERRLOG5	Stores NTTP packet header field User (LSBs) of the logged error	RO	0x0

cs_obs_at_main_ErrorLogger_0_ErrLog7

Module Instance	Base Address	Register Address
i_noc_mpu_m0_cs_obs_at_main_ErrorLogger_0	0xFFD14980	0xFFD149B0

Offset: 0x30

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													ERRLOG7 RO 0x0		

cs_obs_at_main_ErrorLogger_0_ErrLog7 Fields

Bit	Name	Description	Access	Reset
2:0	ERRLOG7	Stores NTTP packet header field Security of the logged error	RO	0x0

noc_mpu_m0_Probe_MPU_main_Probe Address Map

Module Instance	Base Address	End Address
i_noc_mpu_m0_Probe_MPU_main_Probe	0xFFD15000	0xFFD15FFF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
Probe_MPU_main_Probe_Id_CoreId on page 7-629	0x0	32	RO	0xC7360B06	
Probe_MPU_main_Probe_Id_RevisionId on page 7-630	0x4	32	RO	0x129FF00	
Probe_MPU_main_Probe_MainCtl on page 7-631	0x8	32	RW	0x0	Register MainCtl contains probe global control bits. The register has seven bit fields:

Register	Offset	Width	Access	Reset Value	Description
Probe_MPU_main_Probe_CfgCtl on page 7-633	0xC	32	RW	0x0	
Probe_MPU_main_Probe_FilterLut on page 7-634	0x14	32	RW	0x0	
Probe_MPU_main_Probe_TraceAlarmEn on page 7-635	0x18	32	RW	0x0	
Probe_MPU_main_Probe_TraceAlarmStatus on page 7-636	0x1C	32	RO	0x0	
Probe_MPU_main_Probe_TraceAlarmClr on page 7-637	0x20	32	RW	0x0	
Probe_MPU_main_Probe_StatPeriod on page 7-638	0x24	32	RW	0x0	
Probe_MPU_main_Probe_StatGo on page 7-639	0x28	32	RW	0x0	
Probe_MPU_main_Probe_StatAlarmMin on page 7-640	0x2C	32	RW	0x0	
Probe_MPU_main_Probe_StatAlarmMax on page 7-641	0x30	32	RW	0x0	
Probe_MPU_main_Probe_StatAlarmStatus on page 7-642	0x34	32	RO	0x0	
Probe_MPU_main_Probe_StatAlarmClr on page 7-643	0x38	32	RW	0x0	
Probe_MPU_main_Probe_StatAlarmEn on page 7-644	0x3C	32	RW	0x1	
Probe_MPU_main_Probe_Filters_0_RouteIdBase on page 7-645	0x44	32	RW	0x0	

Register	Offset	Width	Access	Reset Value	Description
Probe_MPU_main_Probe_Filters_0_RouteIdMask on page 7-646	0x48	32	RW	0x0	
Probe_MPU_main_Probe_Filters_0_AddrBase_Low on page 7-646	0x4C	32	RW	0x0	
Probe_MPU_main_Probe_Filters_0_WindowSize on page 7-647	0x54	32	RW	0x0	
Probe_MPU_main_Probe_Filters_0_Security-Base on page 7-648	0x58	32	RW	0x0	
Probe_MPU_main_Probe_Filters_0_Security-Mask on page 7-649	0x5C	32	RW	0x0	
Probe_MPU_main_Probe_Filters_0_Opcode on page 7-650	0x60	32	RW	0x0	Packet Probe register Opcode is a 4-bit register that selects candidate packets based on packet opcodes (0 disables the filter):
Probe_MPU_main_Probe_Filters_0_Status on page 7-650	0x64	32	RW	0x0	Register Status is 2-bit register that selects candidate packets based on packet status.
Probe_MPU_main_Probe_Filters_0_Length on page 7-651	0x68	32	RW	0x0	
Probe_MPU_main_Probe_Filters_0_Urgency on page 7-652	0x6C	32	RW	0x0	

Register	Offset	Width	Access	Reset Value	Description
Probe_MPU_main_Probe_Counters_0_Src on page 7-653	0x138	32	RW	0x0	Register CntSrc indicates the event source used to increment the counter. Unassigned values (non-existing Press level or ExtEvent index, or unimplemented Filter) are equivalent to OFF.
Probe_MPU_main_Probe_Counters_0_AlarmMode on page 7-654	0x13C	32	RW	0x0	
Probe_MPU_main_Probe_Counters_0_Val on page 7-654	0x140	32	RO	0x0	
Probe_MPU_main_Probe_Counters_1_Src on page 7-655	0x14C	32	RW	0x0	Register CntSrc indicates the event source used to increment the counter. Unassigned values (non-existing Press level or ExtEvent index, or unimplemented Filter) are equivalent to OFF.
Probe_MPU_main_Probe_Counters_1_AlarmMode on page 7-656	0x150	32	RW	0x0	
Probe_MPU_main_Probe_Counters_1_Val on page 7-657	0x154	32	RO	0x0	
Probe_MPU_main_Probe_Counters_2_Src on page 7-658	0x160	32	RW	0x0	Register CntSrc indicates the event source used to increment the counter. Unassigned values (non-existing Press level or ExtEvent index, or unimplemented Filter) are equivalent to OFF.

Register	Offset	Width	Access	Reset Value	Description
Probe_MPU_main_Probe_Counters_2_AlarmMode on page 7-659	0x164	32	RW	0x0	
Probe_MPU_main_Probe_Counters_2_Val on page 7-660	0x168	32	RO	0x0	
Probe_MPU_main_Probe_Counters_3_Src on page 7-660	0x174	32	RW	0x0	Register CntSrc indicates the event source used to increment the counter. Unassigned values (non-existing Press level or ExtEvent index, or unimplemented Filter) are equivalent to OFF.
Probe_MPU_main_Probe_Counters_3_AlarmMode on page 7-661	0x178	32	RW	0x0	
Probe_MPU_main_Probe_Counters_3_Val on page 7-662	0x17C	32	RO	0x0	

noc_mpu_m0_Probe_MPU_main_Probe Summary

Base Address: 0xFFD15000

Register Address Offset	Bit Fields															
i_noc_mpu_m0_Probe_MPU_main_Probe																
Probe_MPU_main_Probe_Id_CoreId 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	CORECHECKSUM RO 0xC7360B															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CORECHECKSUM RO 0xC7360B								CORETYPEID RO 0x6							

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Probe_MPU_main_Probe_Id_RevisionId 0x4	FLEXNOCID RO 0x129FF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FLEXNOCID RO 0x129FF								USERID RO 0x0							
Probe_MPU_main_Probe_MainCtl 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								FILTBYTEALWAYSCHAINABLEEN RW 0x0	INTRUSIVEMODE RO 0x0	STATCONDDUMP RW 0x0	ALARMEN RW 0x0	STATEN RW 0x0	PAYLOADEN RW 0x0	TRACEEN RW 0x0	ERREN RW 0x0
Probe_MPU_main_Probe_CfgCtl 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														ACTIVE RO 0x0	GLOBALEN RW 0x0
Probe_MPU_main_Probe_FilterLut 0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														FILTERLUT RW 0x0	
Probe_MPU_main_Probe_TraceAlarmEn 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														TRACEALARMEN RW 0x0	
Probe_MPU_main_Probe_TraceAlarm-Status 0x1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														TRACEALARM-STATUS RO 0x0	

Register Address Offset	Bit Fields															
Probe_MPU_main_Probe_TraceAlarmClr 0x20	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														TRACEALARMCLR RW 0x0	
Probe_MPU_main_Probe_StatPeriod 0x24	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved											STATPERIOD RW 0x0				
Probe_MPU_main_Probe_StatGo 0x28	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															STATGO RW 0x0
Probe_MPU_main_Probe_StatAlarmMin 0x2C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	STATALARMMIN RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	STATALARMMIN RW 0x0															
Probe_MPU_main_Probe_StatAlarmMax 0x30	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	STATALARMMAX RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	STATALARMMAX RW 0x0															
Probe_MPU_main_Probe_StatAlarm-Status 0x34	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															STATALARMSTATUS RO 0x0

Register Address Offset	Bit Fields															
Probe_MPU_main_Probe_StatAlarmClr 0x38	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															STATALARMCLR RW 0x0
Probe_MPU_main_Probe_StatAlarmEn 0x3C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															STATALARMEN RW 0x1
Probe_MPU_main_Probe_Filters_0_RouteIdBase 0x44	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved													FILTERS_0_ROUTEIDBASE RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FILTERS_0_ROUTEIDBASE RW 0x0															
Probe_MPU_main_Probe_Filters_0_RouteIdMask 0x48	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved													FILTERS_0_ROUTEIDMASK RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FILTERS_0_ROUTEIDMASK RW 0x0															
Probe_MPU_main_Probe_Filters_0_AddrBase_Low 0x4C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	FILTERS_0_ADDRBASE_LOW RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FILTERS_0_ADDRBASE_LOW RW 0x0															
Probe_MPU_main_Probe_Filters_0_WindowSize 0x54	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved										FILTERS_0_WINDOWSIZE RW 0x0					

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Probe_MPU_main_Probe_Filters_0_SecurityBase 0x58	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													FILTERS_0_SECURITY-BASE RW 0x0		
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Probe_MPU_main_Probe_Filters_0_SecurityMask 0x5C	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													FILTERS_0_SECURITY-MASK RW 0x0		
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Probe_MPU_main_Probe_Filters_0_Opcode 0x60	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved												URGEN RW 0x0	LOCKEN RW 0x0	WREN RW 0x0	RDEN RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Probe_MPU_main_Probe_Filters_0_Status 0x64	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														RSPEN RW 0x0	REQEN RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Probe_MPU_main_Probe_Filters_0_Length 0x68	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved												FILTERS_0_LENGTH RW 0x0			
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Probe_MPU_main_Probe_Filters_0_Urgency 0x6C	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														FILTERS_0_URGENCY RW 0x0	

Register Address Offset	Bit Fields															
Probe_MPU_main_Probe_Counters_0_Src 0x138	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved											INTEVENT RW 0x0				
Probe_MPU_main_Probe_Counters_0_AlarmMode 0x13C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														COUNTERS_0_ALARMMODE RW 0x0	
Probe_MPU_main_Probe_Counters_0_Val 0x140	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	COUNTERS_0_VAL RO 0x0															
Probe_MPU_main_Probe_Counters_1_Src 0x14C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved											INTEVENT RW 0x0				
Probe_MPU_main_Probe_Counters_1_AlarmMode 0x150	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														COUNTERS_1_ALARMMODE RW 0x0	
Probe_MPU_main_Probe_Counters_1_Val 0x154	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	COUNTERS_1_VAL RO 0x0															
Probe_MPU_main_Probe_Counters_2_Src 0x160	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved											INTEVENT RW 0x0				

Register Address Offset	Bit Fields															
Probe_MPU_main_Probe_Counters_2_AlarmMode 0x164	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														COUNTERS_2_ALARMMODE RW 0x0	
Probe_MPU_main_Probe_Counters_2_Val 0x168	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	COUNTERS_2_VAL RO 0x0															
Probe_MPU_main_Probe_Counters_3_Src 0x174	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved											INTEVENT RW 0x0				
Probe_MPU_main_Probe_Counters_3_AlarmMode 0x178	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														COUNTERS_3_ALARMMODE RW 0x0	
Probe_MPU_main_Probe_Counters_3_Val 0x17C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	COUNTERS_3_VAL RO 0x0															

Probe_MPU_main_Probe_Id_CoreId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_MPU_main_Probe	0xFFD15000	0xFFD15000

Offset: 0x0

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CORECHECKSUM RO 0xC7360B															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CORECHECKSUM RO 0xC7360B								CORETYPEID RO 0x6							

Probe_MPU_main_Probe_Id_CoreId Fields

Bit	Name	Description	Access	Reset
31:8	CORECHECKSUM	Field containing a checksum of the parameters of the IP.	RO	0xC7360B
7:0	CORETYPEID	Field identifying the type of IP.	RO	0x6

Probe_MPU_main_Probe_Id_RevisionId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_MPU_main_Probe	0xFFD15000	0xFFD15004

Offset: 0x4

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLEXNOCID RO 0x129FF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLEXNOCID RO 0x129FF								USERID RO 0x0							

Probe_MPU_main_Probe_Id_RevisionId Fields

Bit	Name	Description	Access	Reset
31:8	FLEXNOCID	Field containing the build revision of the software used to generate the IP HDL code.	RO	0x129FF
7:0	USERID	Field containing a user defined value, not used anywhere inside the IP itself.	RO	0x0

Probe_MPU_main_Probe_MainCtl

Register MainCtl contains probe global control bits. The register has seven bit fields:

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_MPU_main_Probe	0xFFD15000	0xFFD15008

Offset: 0x8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								FILTBYTEALWAYSCHAINABLEEN RW 0x0	INTRUSIVEMODE RO 0x0	STATCONDDUMP RW 0x0	ALARMEN RW 0x0	STATEN RW 0x0	PAYLOADEN RW 0x0	TRACEEN RW 0x0	ERREN RW 0x0

Probe_MPU_main_Probe_MainCtl Fields

Bit	Name	Description	Access	Reset
7	FILTBYTEALWAYSCHAINABLEEN	When set to 0, filters are mapped to all statistic counters when counting bytes or enabled bytes. Therefore, only filter events mapped to even counters can be counted using a pair of chained counters.When set to 1, filters are mapped only to even statistic counters when counting bytes or enabled bytes. Thus events from any filter can be counted using a pair of chained counters.	RW	0x0
6	INTRUSIVEMODE	When set to 1, register field IntrusiveMode enables trace operation in Intrusive flow-control mode. When set to 0, the register enables trace operation in Overflow flow-control mode	RO	0x0
5	STATCONDDUMP	When set, register field StatCond-Dump enables the dump of a statistics frame to the range of counter values set for registers StatAlarmMin, StatAlarmMax, and AlarmMode. This field also renders register StatAlarmStatus inopera-tive. When parameter statisticsCoun-terAlarm is set to False, the StatCondDump register bit is reserved.	RW	0x0
4	ALARMEN	When set, register field AlarmEn enables the probe to collect alarm-related information. When the register field bit is null, both TraceAlarm and StatAlarm outputs are driven to 0.	RW	0x0
3	STATEN	When set to 1, register field StatEn enables statistics profiling. The probe sendS statistics results to the output for signal ObsTx. All statistics counters are cleared when the StatEn bit goes from 0 to 1. When set to 0, counters are disabled.	RW	0x0

Bit	Name	Description	Access	Reset
2	PAYLOADEN	Register field PayloadEn, when set to 1, enables traces to contain headers and payload. When set ot 0, only headers are reported.	RW	0x0
1	TRACEEN	Register field TraceEn enables the probe to send filtered packets (Trace) on the ObsTx observation output.	RW	0x0
0	ERREN	Register field ErrEn enables the probe to send on the ObsTx output any packet with Error status, independently of filtering mechanisms, thus constituting a simple supplementary global filter.	RW	0x0

Probe_MPU_main_Probe_CfgCtl

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_MPU_main_Probe	0xFFD15000	0xFFD1500C

Offset: 0xC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														ACTIVE RO 0x0	GLOBALEN RW 0x0

Probe_MPU_main_Probe_CfgCtl Fields

Bit	Name	Description	Access	Reset
1	ACTIVE		RO	0x0
0	GLOBALEN		RW	0x0

Probe_MPU_main_Probe_FilterLut

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_MPU_main_Probe	0xFFD15000	0xFFD15014

Offset: 0x14

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														FILTERLUT RW 0x0	

Probe_MPU_main_Probe_FilterLut Fields

Bit	Name	Description	Access	Reset
1:0	FILTERLUT	Register FilterLut contains a look-up table that is used to combine filter outputs in order to trace packets. Packet tracing is enabled when the FilterLut bit of index (FNout ... F0out) is equal to 1.The number of bits in register FilterLut is determined by the setting for parameter nFilter, calculated as 2**nFilter.When parameter nFilter is set to None, FilterLut is reserved.	RW	0x0

Probe_MPU_main_Probe_TraceAlarmEn

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_MPU_main_Probe	0xFFD15000	0xFFD15018

Offset: 0x18

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														TRACEALARMEN RW 0x0	

Probe_MPU_main_Probe_TraceAlarmEn Fields

Bit	Name	Description	Access	Reset
1:0	TRACEALARMEN	Register TraceAlarmEn controls which lookup table or filter can set the TraceAlarm signal output once the trace alarm status is set. The number of bits in register TraceAlarmEn is determined by the value set for parameter nFilter + 1.Bit nFilter controls the lookup table output, and bits nFilter:0 control the corresponding filter output. When parameter nFilter is set to None, TraceAlarmEn is reserved.	RW	0x0

Probe_MPU_main_Probe_TraceAlarmStatus

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_MPU_main_Probe	0xFFD15000	0xFFD1501C

Offset: 0x1C

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														TRACEALARM-STATUS RO 0x0	

Probe_MPU_main_Probe_TraceAlarmStatus Fields

Bit	Name	Description	Access	Reset
1:0	TRACEALARMSTATUS	Register TraceAlarmStatus is a read-only register that indicates which lookup table or filter has been matched by a packet, independently of register TraceAlarmEn bit configuration. The number of bits in TraceAlarmStatus is determined by the value set for parameter nFilter + 1.When nFilter is set to None, TraceAlarmStatus is reserved.	RO	0x0

Probe_MPU_main_Probe_TraceAlarmClr

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_MPU_main_Probe	0xFFD15000	0xFFD15020

Offset: 0x20

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														TRACEALARMCLR RW 0x0	

Probe_MPU_main_Probe_TraceAlarmClr Fields

Bit	Name	Description	Access	Reset
1:0	TRACEALARMCLR	Setting a bit to 1 in register TraceAlarmClr clears the corresponding bit in register TraceAlarmStatus.The number of bits in register TraceAlarmClr is equal to (nFilter + 1). When nFilter is set to 0, TraceAlarmClr is reserved.NOTE The written value is not stored in TraceAlarmClr. A read always returns 0.	RW	0x0

Probe_MPU_main_Probe_StatPeriod

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_MPU_main_Probe	0xFFD15000	0xFFD15024

Offset: 0x24

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved											STATPERIOD RW 0x0				

Probe_MPU_main_Probe_StatPeriod Fields

Bit	Name	Description	Access	Reset
4:0	STATPERIOD	Register StatPeriod is a 5-bit register that sets a period, within a range of 2 cycles to 2 gigacycles, during which statistics are collected before being dumped automatically. Setting the register implicitly enables automatic mode operation for statistics collection. The period is calculated with the formula: N_Cycle = 2**StatPeriodWhen register StatPeriod is set to its default value 0, automatic dump mode is disabled, and register StatGo is activated for manual mode operation. Note: When parameter statisticsCollection is set to False, StatPeriod is reserved.	RW	0x0

Probe_MPU_main_Probe_StatGo

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_MPU_main_Probe	0xFFD15000	0xFFD15028

Offset: 0x28

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															STATGO RW 0x0

Probe_MPU_main_Probe_StatGo Fields

Bit	Name	Description	Access	Reset
0	STATGO	Writing a 1 to the 1-bit pulse register StatGo generates a statistics dump.The register is active when statistics collection operates in manual mode, that is, when register StatPeriod is set to 0.NOTE The written value is not stored in StatGo. A read always returns 0.	RW	0x0

Probe_MPU_main_Probe_StatAlarmMin

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_MPU_main_Probe	0xFFD15000	0xFFD1502C

Offset: 0x2C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
STATALARMMIN RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STATALARMMIN RW 0x0															

Probe_MPU_main_Probe_StatAlarmMin Fields

Bit	Name	Description	Access	Reset
31:0	STATALARMMIN	Register StatAlarmMin contains the minimum count value used in statistics alarm comparisons. The number of bits is equal to twice the value set forparameter wStatisticsCounter. When parameter statisticsCounterAlarm is set to False, StatAlarmMin is reserved.	RW	0x0

Probe_MPU_main_Probe_StatAlarmMax

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_MPU_main_Probe	0xFFD15000	0xFFD15030

Offset: 0x30

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
STATALARMMAX RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STATALARMMAX RW 0x0															

Probe_MPU_main_Probe_StatAlarmMax Fields

Bit	Name	Description	Access	Reset
31:0	STATALARMMAX	Register StatAlarmMax contains the maximum count value used in statistics alarm comparisons.The number of bits is equal to twice the value set for parameter wStatisticsCounter. When parameter statisticsCounterAlarm is set to False, StatAlarmMax is reserved.	RW	0x0

Probe_MPU_main_Probe_StatAlarmStatus

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_MPU_main_Probe	0xFFD15000	0xFFD15034

Offset: 0x34

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															STATALARMSTATUS RO 0x0

Probe_MPU_main_Probe_StatAlarmStatus Fields

Bit	Name	Description	Access	Reset
0	STATALARMSTATUS	Register StatAlarmStatus is a read-only 1-bit register indicating that at least one statistics counter has exceeded the programmed values for registers StatAlarmMin or StatAlarmMax. Output signal StatAlarm is equal to the values stored in register MainCtl fields StatAlarmStatus and AlarmEn. When parameter statisticsCounterAlarm is set to False, StatAlarmStatus is reserved.	RO	0x0

Probe_MPU_main_Probe_StatAlarmClr

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_MPU_main_Probe	0xFFD15000	0xFFD15038

Offset: 0x38

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															STATALARMCLR RW 0x0

Probe_MPU_main_Probe_StatAlarmClr Fields

Bit	Name	Description	Access	Reset
0	STATALARMCLR	Register StatAlarmClr is a 1-bit register. Writing a 1 to this register clears the StatAlarmStatus register bit.When parameter statisticsCounterAlarm is set to False, StatAlarmClr is reserved.NOTE The written value is not stored in StatAlarmClr. A read always returns 0.	RW	0x0

Probe_MPU_main_Probe_StatAlarmEn

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_MPU_main_Probe	0xFFD15000	0xFFD1503C

Offset: 0x3C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															STATALARMEN RW 0x1

Probe_MPU_main_Probe_StatAlarmEn Fields

Bit	Name	Description	Access	Reset
0	STATALARMEN	Register StatAlarmEn is a 1-bit register. When set to 0 it masks StatAlarm and CtiTrigOut(1) signal interrupts.	RW	0x1

Probe_MPU_main_Probe_Filters_0_RouteIdBase

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_MPU_main_Probe	0xFFD15000	0xFFD15044

Offset: 0x44

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													FILTERS_0_ROUTEIDBASE RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FILTERS_0_ROUTEIDBASE RW 0x0															

Probe_MPU_main_Probe_Filters_0_RouteIdBase Fields

Bit	Name	Description	Access	Reset
18:0	FILTERS_0_ROUTEIDBASE	Register RouteIdBase contains the RouteId-lsbFilterRouteId bits base used to filter packets.	RW	0x0

Probe_MPU_main_Probe_Filters_0_RouteIdMask

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_MPU_main_Probe	0xFFD15000	0xFFD15048

Offset: 0x48

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													FILTERS_0_ROUTEIDMASK RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FILTERS_0_ROUTEIDMASK RW 0x0															

Probe_MPU_main_Probe_Filters_0_RouteIdMask Fields

Bit	Name	Description	Access	Reset
18:0	FILTERS_0_ROUTEIDMASK	Register RouteIdMask contains the RouteId-lsbFilterRouteId mask used to filter packets. A packet is a candidate when packet.RouteId>> lsbFilterRouteId & RouteIdMask = RouteIdBase & RouteIdMask.	RW	0x0

Probe_MPU_main_Probe_Filters_0_AddrBase_Low

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_MPU_main_Probe	0xFFD15000	0xFFD1504C

Offset: 0x4C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FILTERS_0_ADDRBASE_LOW RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FILTERS_0_ADDRBASE_LOW RW 0x0															

Probe_MPU_main_Probe_Filters_0_AddrBase_Low Fields

Bit	Name	Description	Access	Reset
31:0	FILTERS_0_ADDRBASE_LOW	Address LSB register.	RW	0x0

Probe_MPU_main_Probe_Filters_0_WindowSize

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_MPU_main_Probe	0xFFD15000	0xFFD15054

Offset: 0x54

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										FILTERS_0_WINDOWSIZE RW 0x0					

Probe_MPU_main_Probe_Filters_0_WindowSize Fields

Bit	Name	Description	Access	Reset
5:0	FILTERS_0_WINDOWSIZE	Register WindowSize contains the encoded address mask used to filter packets. The effective Mask value is equal to ~(2max(WindowSize, packet.Len) - 1). A packet is a candidate when packet.Addr & Mask = AddrBase & Mask. This allows filteringof packets having an intersection with the AddrBase/ WindowSize burst aligned region, even if the region is smaller than the packet.	RW	0x0

Probe_MPU_main_Probe_Filters_0_SecurityBase

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_MPU_main_Probe	0xFFD15000	0xFFD15058

Offset: 0x58

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													FILTERS_0_SECURITY-BASE RW 0x0		

Probe_MPU_main_Probe_Filters_0_SecurityBase Fields

Bit	Name	Description	Access	Reset
2:0	FILTERS_0_SECURITYBASE	Register SecurityBase contains the security base used to filter packets.	RW	0x0

Probe_MPU_main_Probe_Filters_0_SecurityMask

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_MPU_main_Probe	0xFFD15000	0xFFD1505C

Offset: 0x5C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													FILTERS_0_SECURITY-MASK RW 0x0		

Probe_MPU_main_Probe_Filters_0_SecurityMask Fields

Bit	Name	Description	Access	Reset
2:0	FILTERS_0_SECURITYMASK	Register SecurityMask is contains the security mask used to filter packets. A packet is a candidate when: packet.Security & Security-Mask = SecurityBase & SecurityMasks.	RW	0x0

Probe_MPU_main_Probe_Filters_0_Opcode

Packet Probe register Opcode is a 4-bit register that selects candidate packets based on packet opcodes (0 disables the filter):

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_MPU_main_Probe	0xFFD15000	0xFFD15060

Offset: 0x60

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												URGEN RW 0x0	LOCKEN RW 0x0	WREN RW 0x0	RDEN RW 0x0

Probe_MPU_main_Probe_Filters_0_Opcode Fields

Bit	Name	Description	Access	Reset
3	URGEN	Selects URG packets (urgency).	RW	0x0
2	LOCKEN	Selects RDX-WR, RDL, WRC and Linked sequence.	RW	0x0
1	WREN	Selects WR packets.	RW	0x0
0	RDEN	Selects RD packets.	RW	0x0

Probe_MPU_main_Probe_Filters_0_Status

Register Status is 2-bit register that selects candidate

packets based on packet status.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_MPU_main_Probe	0xFFD15000	0xFFD15064

Offset: 0x64

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														RSPEN RW 0x0	REQEN RW 0x0

Probe_MPU_main_Probe_Filters_0_Status Fields

Bit	Name	Description	Access	Reset
1	RSPEN	Selects RSP and FAIL-CONT status packets.	RW	0x0
0	REQEN	Selects REQ status packets.	RW	0x0

Probe_MPU_main_Probe_Filters_0_Length

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_MPU_main_Probe	0xFFD15000	0xFFD15068

Offset: 0x68

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												FILTERS_0_LENGTH RW 0x0			

Probe_MPU_main_Probe_Filters_0_Length Fields

Bit	Name	Description	Access	Reset
3:0	FILTERS_0_LENGTH	Register Length is 4-bit register that selects candidate packets if their number of bytes is less than or equal to 2**Length.	RW	0x0

Probe_MPU_main_Probe_Filters_0_Urgency

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_MPU_main_Probe	0xFFD15000	0xFFD1506C

Offset: 0x6C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														FILTERS_0_URGENCY RW 0x0	

Probe_MPU_main_Probe_Filters_0_Urgency Fields

Bit	Name	Description	Access	Reset
1:0	FILTERS_0_URGENCY	Register Urgency contains the minimum urgency level used to filter packets. A packet is a candidate when its socket urgency is greater than or equal to the urgency specified in the register.	RW	0x0

Probe_MPU_main_Probe_Counters_0_Src

Register CntSrc indicates the event source used to increment the counter. Unassigned values (non-existing Press level or ExtEvent index, or unimplemented Filter) are equivalent to OFF.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_MPU_main_Probe	0xFFD15000	0xFFD15138

Offset: 0x138

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved											INTEVENT RW 0x0				

Probe_MPU_main_Probe_Counters_0_Src Fields

Bit	Name	Description	Access	Reset
4:0	INTEVENT	Internal packet event	RW	0x0

Probe_MPU_main_Probe_Counters_0_AlarmMode

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_MPU_main_Probe	0xFFD15000	0xFFD1513C

Offset: 0x13C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														COUNTERS_0_ALARMMODE RW 0x0	

Probe_MPU_main_Probe_Counters_0_AlarmMode Fields

Bit	Name	Description	Access	Reset
1:0	COUNTERS_0_ALARMMODE	Register AlarmMode is a 2-bit register that is present when parameter statisticsCounterAlarm is set to True. The register defines the statistics-alarm behavior of the counter.	RW	0x0

Probe_MPU_main_Probe_Counters_0_Val

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_MPU_main_Probe	0xFFD15000	0xFFD15140

Offset: 0x140

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COUNTERS_0_VAL RO 0x0															

Probe_MPU_main_Probe_Counters_0_Val Fields

Bit	Name	Description	Access	Reset
15:0	COUNTERS_0_VAL	Register Val is a read-only register that is always present. The register containsthe statistics counter value either pending StatAlarm output, or when statisticscollection is suspended subsequent to triggers or signal statSuspend.	RO	0x0

Probe_MPU_main_Probe_Counters_1_Src

Register CntSrc indicates the event source used to increment the counter. Unassigned values (non-existing Press level or ExtEvent index, or unimplemented Filter) are equivalent to OFF.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_MPU_main_Probe	0xFFD15000	0xFFD1514C

Offset: 0x14C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved											INTEVENT RW 0x0				

Probe_MPU_main_Probe_Counters_1_Src Fields

Bit	Name	Description	Access	Reset
4:0	INTEVENT	Internal packet event	RW	0x0

Probe_MPU_main_Probe_Counters_1_AlarmMode

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_MPU_main_Probe	0xFFD15000	0xFFD15150

Offset: 0x150

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														COUNTERS_1_ALARMMODE RW 0x0	

Probe_MPU_main_Probe_Counters_1_AlarmMode Fields

Bit	Name	Description	Access	Reset
1:0	COUNTERS_1_ALARMMODE	Register AlarmMode is a 2-bit register that is present when parameter statisticsCounterAlarm is set to True. The register defines the statistics-alarm behavior of the counter.	RW	0x0

Probe_MPU_main_Probe_Counters_1_Val

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_MPU_main_Probe	0xFFD15000	0xFFD15154

Offset: 0x154

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COUNTERS_1_VAL RO 0x0															

Probe_MPU_main_Probe_Counters_1_Val Fields

Bit	Name	Description	Access	Reset
15:0	COUNTERS_1_VAL	Register Val is a read-only register that is always present. The register containsthe statistics counter value either pending StatAlarm output, or when statisticscollection is suspended subsequent to triggers or signal statSuspend.	RO	0x0

Probe_MPU_main_Probe_Counters_2_Src

Register CntSrc indicates the event source used to increment the counter. Unassigned values (non-existing Press level or ExtEvent index, or unimplemented Filter) are equivalent to OFF.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_MPU_main_Probe	0xFFD15000	0xFFD15160

Offset: 0x160

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved											INTEVENT RW 0x0				

Probe_MPU_main_Probe_Counters_2_Src Fields

Bit	Name	Description	Access	Reset
4:0	INTEVENT	Internal packet event	RW	0x0

Probe_MPU_main_Probe_Counters_2_AlarmMode

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_MPU_main_Probe	0xFFD15000	0xFFD15164

Offset: 0x164

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														COUNTERS_2_ALARMMODE RW 0x0	

Probe_MPU_main_Probe_Counters_2_AlarmMode Fields

Bit	Name	Description	Access	Reset
1:0	COUNTERS_2_ALARMMODE	Register AlarmMode is a 2-bit register that is present when parameter statisticsCounterAlarm is set to True. The register defines the statistics-alarm behavior of the counter.	RW	0x0

Probe_MPU_main_Probe_Counters_2_Val

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_MPU_main_Probe	0xFFD15000	0xFFD15168

Offset: 0x168

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COUNTERS_2_VAL RO 0x0															

Probe_MPU_main_Probe_Counters_2_Val Fields

Bit	Name	Description	Access	Reset
15:0	COUNTERS_2_VAL	Register Val is a read-only register that is always present. The register containsthe statistics counter value either pending StatAlarm output, or when statisticscollection is suspended subsequent to triggers or signal statSuspend.	RO	0x0

Probe_MPU_main_Probe_Counters_3_Src

Register CntSrc indicates the event source used to increment the counter. Unassigned values (non-existing Press level or ExtEvent index, or unimplemented Filter) are equivalent to OFF.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_MPU_main_Probe	0xFFD15000	0xFFD15174

Offset: 0x174

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved											INTEVENT RW 0x0				

Probe_MPU_main_Probe_Counters_3_Src Fields

Bit	Name	Description	Access	Reset
4:0	INTEVENT	Internal packet event	RW	0x0

Probe_MPU_main_Probe_Counters_3_AlarmMode

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_MPU_main_Probe	0xFFD15000	0xFFD15178

Offset: 0x178

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														COUNTERS_3_ALARMMODE RW 0x0	

Probe_MPU_main_Probe_Counters_3_AlarmMode Fields

Bit	Name	Description	Access	Reset
1:0	COUNTERS_3_ALARMMODE	Register AlarmMode is a 2-bit register that is present when parameter statisticsCounterAlarm is set to True. The register defines the statistics-alarm behavior of the counter.	RW	0x0

Probe_MPU_main_Probe_Counters_3_Val

Module Instance	Base Address	Register Address
i_noc_mpu_m0_Probe_MPU_main_Probe	0xFFD15000	0xFFD1517C

Offset: 0x17C

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COUNTERS_3_VAL RO 0x0															

Probe_MPU_main_Probe_Counters_3_Val Fields

Bit	Name	Description	Access	Reset
15:0	COUNTERS_3_VAL	Register Val is a read-only register that is always present. The register containsthe statistics counter value either pending StatAlarm output, or when statisticscollection is suspended subsequent to triggers or signal statSuspend.	RO	0x0

noc_mpu_m0_I_main_QosGenerator Address Map

Module Instance	Base Address	End Address
i_noc_mpu_m0_mpu_m0_I_main_QosGenerator	0xFFD16000	0xFFD1607F

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
mpu_m0_I_main_QosGenerator_Id_CoreId on page 7-665	0x0	32	RO	0x30ACDD04	
mpu_m0_I_main_QosGenerator_Id_RevisionId on page 7-666	0x4	32	RO	0x129FF00	
mpu_m0_I_main_QosGenerator_Priority on page 7-667	0x8	32	RW	0x80000202	Priority register.
mpu_m0_I_main_QosGenerator_Mode on page 7-668	0xC	32	RW	0x1	
mpu_m0_I_main_QosGenerator_Bandwidth on page 7-669	0x10	32	RW	0x6AA	

Register	Offset	Width	Access	Reset Value	Description
mpu_m0_I_main_QosGenerator_Saturation on page 7-670	0x14	32	RW	0x8	
mpu_m0_I_main_QosGenerator_ExtControl on page 7-671	0x18	32	RW	0x0	External inputs control.

noc_mpu_m0_l_main_QosGenerator Summary

Base Address: 0xFFD16000

Register Address Offset	Bit Fields															
i_noc_mpu_m0_mpu_m0_I_main_QosGener-ator																
mpu_m0_I_main_QosGenerator_Id_CoreId 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	CORECHECKSUM RO 0x30ACDD															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CORECHECKSUM RO 0x30ACDD								CORETYPEID RO 0x4							
mpu_m0_I_main_QosGenerator_Id_RevisionId 0x4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	FLEXNOCID RO 0x129FF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FLEXNOCID RO 0x129FF								USERID RO 0x0							
mpu_m0_I_main_QosGenerator_Priority 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	MARK RO 0x1	Reserved														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						P1 RW 0x2		Reserved						P0 RW 0x2	

Register Address Offset	Bit Fields															
mpu_m0_I_main_QosGenerator_Mode 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														MODE RW 0x1	
mpu_m0_I_main_QosGenerator_Bandwidth 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				BANDWIDTH RW 0x6AA											
mpu_m0_I_main_QosGenerator_Saturation 0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						SATURATION RW 0x8									
mpu_m0_I_main_QosGenerator_ExtControl 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													INTCLKEN RW 0x0	EXTTHREN RW 0x0	SOCKETQOSEN RW 0x0

mpu_m0_I_main_QosGenerator_Id_CoreId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_mpu_m0_I_main_QosGenerator	0xFFD16000	0xFFD16000

Offset: 0x0

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CORECHECKSUM RO 0x30ACDD															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CORECHECKSUM RO 0x30ACDD								CORETYPEID RO 0x4							

mpu_m0_l_main_QosGenerator_Id_CoreId Fields

Bit	Name	Description	Access	Reset
31:8	CORECHECKSUM	Field containing a checksum of the parameters of the IP.	RO	0x30ACDD
7:0	CORETYPEID	Field identifying the type of IP.	RO	0x4

mpu_m0_l_main_QosGenerator_Id_RevisionId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_mpu_m0_I_main_QosGenerator	0xFFD16000	0xFFD16004

Offset: 0x4

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLEXNOCID RO 0x129FF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLEXNOCID RO 0x129FF								USERID RO 0x0							

mpu_m0_l_main_QosGenerator_Id_RevisionId Fields

Bit	Name	Description	Access	Reset
31:8	FLEXNOCID	Field containing the build revision of the software used to generate the IP HDL code.	RO	0x129FF
7:0	USERID	Field containing a user defined value, not used anywhere inside the IP itself.	RO	0x0

mpu_m0_l_main_QosGenerator_Priority

Priority register.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_mpu_m0_I_main_QosGenerator	0xFFD16000	0xFFD16008

Offset: 0x8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MARK RO 0x1	Reserved														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						P1 RW 0x2		Reserved						P0 RW 0x2	

mpu_m0_l_main_QosGenerator_Priority Fields

Bit	Name	Description	Access	Reset
31	MARK	Backward compatibility marker when 0.	RO	0x1

Bit	Name	Description	Access	Reset
9:8	P1	In Programmable or Bandwidth Limiter mode, the priority level for read transactions. In Bandwidth regulator mode, the priority level when the used throughput is below the threshold. In Bandwidth Regulator mode, P1 should have a value equal or greater than P0.	RW	0x2
1:0	P0	In Programmable or Bandwidth Limiter mode, the priority level for write transactions. In Bandwidth Regulator mode, the priority level when the used throughput is above the threshold. In Bandwidth Regulator mode, P0 should have a value equal or lower than P1.	RW	0x2

mpu_m0_l_main_QosGenerator_Mode

Module Instance	Base Address	Register Address
i_noc_mpu_m0_mpu_m0_I_main_QosGenerator	0xFFD16000	0xFFD1600C

Offset: 0xC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														MODE RW 0x1	

mpu_m0_l_main_QosGenerator_Mode Fields

Bit	Name	Description	Access	Reset
1:0	MODE	0 = Programmable mode: a programmed priority is assigned to each read or write, 1 = Bandwidth Limiter Mode: a hard limit restricts throughput, 2 = Bypass mode: (<See SoC-specific QoS generator documentation>), 3 = Bandwidth Regulator mode: priority decreases when throughput exceeds a threshold.	RW	0x1

mpu_m0_l_main_QosGenerator_Bandwidth

Module Instance	Base Address	Register Address
i_noc_mpu_m0_mpu_m0_I_main_QosGenerator	0xFFD16000	0xFFD16010

Offset: 0x10

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				BANDWIDTH RW 0x6AA											

mpu_m0_l_main_QosGenerator_Bandwidth Fields

Bit	Name	Description	Access	Reset
11:0	BANDWIDTH	In Bandwidth Limiter or Bandwidth Regulator mode, the bandwidth threshold is in units of 1/256th bytes per cycle. The formula to calculate this value is: Bandwidth = Throughput (Mbsp) * 256 / Clock Frequency. For example, 80 MBps on a 250 MHz interface has the value 0x0052 (BW = 80Mbps * 256/ 250MHz = 81.92, which is rounded to 0x0052).	RW	0x6AA

mpu_m0_l_main_QosGenerator_Saturation

Module Instance	Base Address	Register Address
i_noc_mpu_m0_mpu_m0_I_main_QosGenerator	0xFFD16000	0xFFD16014

Offset: 0x14

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						SATURATION RW 0x8									

mpu_m0_l_main_QosGenerator_Saturation Fields

Bit	Name	Description	Access	Reset
9:0	SATURATION	In Bandwidth Limiter or Bandwidth Regulator mode, the maximum data count value, in units of 16 bytes. This determines the window of time over which bandwidth is measured. For example, to measure bandwidth within a 1000 cycle window on a 64-bit interface is value 0x1F4.	RW	0x8

mpu_m0_l_main_QosGenerator_ExtControl

External inputs control.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_mpu_m0_I_main_QosGenerator	0xFFD16000	0xFFD16018

Offset: 0x18

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													INTCLKEN RW 0x0	EXTTHREN RW 0x0	SOCKETQOSEN RW 0x0

mpu_m0_l_main_QosGenerator_ExtControl Fields

Bit	Name	Description	Access	Reset
2	INTCLKEN	n/a	RW	0x0
1	EXTTHREN	n/a	RW	0x0

Bit	Name	Description	Access	Reset
0	SOCKETQOSEN	n/a	RW	0x0

noc_mpu_m1_I_main_QosGenerator Address Map

Module Instance	Base Address	End Address
i_noc_mpu_m0_mpu_m1_I_main_QosGenerator	0xFFD16080	0xFFD160FF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
mpu_m1_I_main_QosGenerator_Id_CoreId on page 7-674	0x0	32	RO	0x84F97E04	
mpu_m1_I_main_QosGenerator_Id_RevisionId on page 7-675	0x4	32	RO	0x129FF00	
mpu_m1_I_main_QosGenerator_Priority on page 7-676	0x8	32	RW	0x80000202	Priority register.
mpu_m1_I_main_QosGenerator_Mode on page 7-677	0xC	32	RW	0x1	
mpu_m1_I_main_QosGenerator_Bandwidth on page 7-678	0x10	32	RW	0x6AA	
mpu_m1_I_main_QosGenerator_Saturation on page 7-679	0x14	32	RW	0x8	
mpu_m1_I_main_QosGenerator_ExtControl on page 7-680	0x18	32	RW	0x0	External inputs control.

noc_mpu_m1_I_main_QosGenerator Summary

Base Address: 0xFFD16080

Register Address Offset	Bit Fields															
i_noc_mpu_m0_mpu_m1_I_main_QosGener-ator																
mpu_m1_I_main_QosGenerator_Id_CoreId 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	CORECHECKSUM RO 0x84F97E															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CORECHECKSUM RO 0x84F97E								CORETYPEID RO 0x4							
mpu_m1_I_main_QosGenerator_Id_RevisionId 0x4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	FLEXNOCID RO 0x129FF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FLEXNOCID RO 0x129FF								USERID RO 0x0							
mpu_m1_I_main_QosGenerator_Priority 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	MARK RO 0x1	Reserved														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						P1 RW 0x2		Reserved						P0 RW 0x2	
mpu_m1_I_main_QosGenerator_Mode 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														MODE RW 0x1	
mpu_m1_I_main_QosGenerator_Bandwidth 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				BANDWIDTH RW 0x6AA											

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
mpu_m1_I_main_QosGenerator_Saturation 0x14	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						SATURATION RW 0x8									
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
mpu_m1_I_main_QosGenerator_ExtControl 0x18	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													INTCLKEN RW 0x0	EXTTHREN RW 0x0	SOCKETQOSEN RW 0x0

mpu_m1_I_main_QosGenerator_Id_CoreId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_mpu_m1_I_main_QosGenerator	0xFFD16080	0xFFD16080

Offset: 0x0

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CORECHECKSUM RO 0x84F97E															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CORECHECKSUM RO 0x84F97E								CORETYPEID RO 0x4							

mpu_m1_I_main_QosGenerator_Id_CoreId Fields

Bit	Name	Description	Access	Reset
31:8	CORECHECKSUM	Field containing a checksum of the parameters of the IP.	RO	0x84F97E
7:0	CORETYPEID	Field identifying the type of IP.	RO	0x4

mpu_m1_I_main_QosGenerator_Id_RevisionId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_mpu_m1_I_main_QosGenerator	0xFFD16080	0xFFD16084

Offset: 0x4

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLEXNOCID RO 0x129FF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLEXNOCID RO 0x129FF								USERID RO 0x0							

mpu_m1_I_main_QosGenerator_Id_RevisionId Fields

Bit	Name	Description	Access	Reset
31:8	FLEXNOCID	Field containing the build revision of the software used to generate the IP HDL code.	RO	0x129FF
7:0	USERID	Field containing a user defined value, not used anywhere inside the IP itself.	RO	0x0

mpu_m1_l_main_QosGenerator_Priority

Priority register.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_mpu_m1_I_main_QosGenerator	0xFFD16080	0xFFD16088

Offset: 0x8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MARK RO 0x1	Reserved														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						P1 RW 0x2		Reserved						P0 RW 0x2	

mpu_m1_l_main_QosGenerator_Priority Fields

Bit	Name	Description	Access	Reset
31	MARK	Backward compatibility marker when 0.	RO	0x1
9:8	P1	In Programmable or Bandwidth Limiter mode, the priority level for read transactions. In Bandwidth regulator mode, the priority level when the used throughput is below the threshold. In Bandwidth Regulator mode, P1 should have a value equal or greater than P0.	RW	0x2

Bit	Name	Description	Access	Reset
1:0	P0	In Programmable or Bandwidth Limiter mode, the priority level for write transactions. In Bandwidth Regulator mode, the priority level when the used throughput is above the threshold. In Bandwidth Regulator mode, P0 should have a value equal or lower than P1.	RW	0x2

mpu_m1_I_main_QosGenerator_Mode

Module Instance	Base Address	Register Address
i_noc_mpu_m0_mpu_m1_I_main_QosGenerator	0xFFD16080	0xFFD1608C

Offset: 0xC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														MODE RW 0x1	

mpu_m1_l_main_QosGenerator_Mode Fields

Bit	Name	Description	Access	Reset
1:0	MODE	0 = Programmable mode: a programmed priority is assigned to each read or write, 1 = Bandwidth Limiter Mode: a hard limit restricts throughput, 2 = Bypass mode: (<See SoC-specific QoS generator documentation>), 3 = Bandwidth Regulator mode: priority decreases when throughput exceeds a threshold.	RW	0x1

mpu_m1_l_main_QosGenerator_Bandwidth

Module Instance	Base Address	Register Address
i_noc_mpu_m0_mpu_m1_I_main_QosGenerator	0xFFD16080	0xFFD16090

Offset: 0x10

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				BANDWIDTH RW 0x6AA											

mpu_m1_I_main_QosGenerator_Bandwidth Fields

Bit	Name	Description	Access	Reset
11:0	BANDWIDTH	In Bandwidth Limiter or Bandwidth Regulator mode, the bandwidth threshold is in units of 1/256th bytes per cycle. The formula to calculate this value is: Bandwidth = Throughput (Mbsp) * 256 / Clock Frequency. For example, 80 MBps on a 250 MHz interface has the value 0x0052 (BW = 80Mbps * 256/ 250MHz = 81.92, which is rounded to 0x0052).	RW	0x6AA

mpu_m1_I_main_QosGenerator_Saturation

Module Instance	Base Address	Register Address
i_noc_mpu_m0_mpu_m1_I_main_QosGenerator	0xFFD16080	0xFFD16094

Offset: 0x14

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						SATURATION RW 0x8									

mpu_m1_I_main_QosGenerator_Saturation Fields

Bit	Name	Description	Access	Reset
9:0	SATURATION	In Bandwidth Limiter or Bandwidth Regulator mode, the maximum data count value, in units of 16 bytes. This determines the window of time over which bandwidth is measured. For example, to measure bandwidth within a 1000 cycle window on a 64-bit interface is value 0x1F4.	RW	0x8

mpu_m1_I_main_QosGenerator_ExtControl

External inputs control.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_mpu_m1_I_main_QosGenerator	0xFFD16080	0xFFD16098

Offset: 0x18

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													INTCLKEN RW 0x0	EXTTHREN RW 0x0	SOCKETQOSEN RW 0x0

mpu_m1_I_main_QosGenerator_ExtControl Fields

Bit	Name	Description	Access	Reset
2	INTCLKEN	n/a	RW	0x0
1	EXTTHREN	n/a	RW	0x0

Bit	Name	Description	Access	Reset
0	SOCKETQOSEN	n/a	RW	0x0

noc_mpu_fpga2soc_axi32_I_main_QosGenerator Address Map

Module Instance	Base Address	End Address
i_noc_mpu_m0_fpga2soc_axi32_I_main_QosGenerator	0xFFD16100	0xFFD1617F

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
fpga2soc_axi32_I_main_QosGenerator_Id_CoreId on page 7-683	0x0	32	RO	0xF012EA04	
fpga2soc_axi32_I_main_QosGenerator_Id_RevisionId on page 7-684	0x4	32	RO	0x129FF00	
fpga2soc_axi32_I_main_QosGenerator_Priority on page 7-685	0x8	32	RW	0x80000202	Priority register.
fpga2soc_axi32_I_main_QosGenerator_Mode on page 7-686	0xC	32	RW	0x1	
fpga2soc_axi32_I_main_QosGenerator_Bandwidth on page 7-687	0x10	32	RW	0x280	
fpga2soc_axi32_I_main_QosGenerator_Saturation on page 7-688	0x14	32	RW	0x8	

Register	Offset	Width	Access	Reset Value	Description
fpga2soc_axi32_I_main_QosGenerator_ExtControl on page 7-689	0x18	32	RW	0x0	External inputs control.

noc_mpu_fpga2soc_axi32_I_main_QosGenerator Summary

Base Address: 0xFFD16100

Register Address Offset	Bit Fields															
i_noc_mpu_m0_fpga2soc_axi32_I_main_QosGenerator																
fpga2soc_axi32_I_main_QosGenerator_Id_CoreId 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	CORECHECKSUM RO 0xF012EA															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CORECHECKSUM RO 0xF012EA								CORETYPEID RO 0x4							
fpga2soc_axi32_I_main_QosGenerator_Id_RevisionId 0x4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	FLEXNOCID RO 0x129FF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FLEXNOCID RO 0x129FF								USERID RO 0x0							
fpga2soc_axi32_I_main_QosGenerator_Priority 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	MARK RO 0x1	Reserved														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						P1 RW 0x2		Reserved						P0 RW 0x2	

Register Address Offset	Bit Fields															
fpga2soc_axi32_I_main_QosGenerator_Mode 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														MODE RW 0x1	
fpga2soc_axi32_I_main_QosGenerator_Bandwidth 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved					BANDWIDTH RW 0x280										
fpga2soc_axi32_I_main_QosGenerator_Saturation 0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						SATURATION RW 0x8									
fpga2soc_axi32_I_main_QosGenerator_ExtControl 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													INTCLKEN RW 0x0	EXTTHREN RW 0x0	SOCKETQOSEN RW 0x0

fpga2soc_axi32_I_main_QosGenerator_Id_CoreId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2soc_axi32_I_main_QosGenerator	0xFFD16100	0xFFD16100

Offset: 0x0

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CORECHECKSUM RO 0xF012EA															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CORECHECKSUM RO 0xF012EA								CORETYPEID RO 0x4							

fpga2soc_axi32_I_main_QosGenerator_Id_CoreId Fields

Bit	Name	Description	Access	Reset
31:8	CORECHECKSUM	Field containing a checksum of the parameters of the IP.	RO	0xF012EA
7:0	CORETYPEID	Field identifying the type of IP.	RO	0x4

fpga2soc_axi32_I_main_QosGenerator_Id_RevisionId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2soc_axi32_I_main_QosGenerator	0xFFD16100	0xFFD16104

Offset: 0x4

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLEXNOCID RO 0x129FF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLEXNOCID RO 0x129FF								USERID RO 0x0							

fpga2soc_axi32_I_main_QosGenerator_Id_RevisionId Fields

Bit	Name	Description	Access	Reset
31:8	FLEXNOCID	Field containing the build revision of the software used to generate the IP HDL code.	RO	0x129FF
7:0	USERID	Field containing a user defined value, not used anywhere inside the IP itself.	RO	0x0

fpga2soc_axi32_I_main_QosGenerator_Priority

Priority register.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2soc_axi32_I_main_QosGenerator	0xFFD16100	0xFFD16108

Offset: 0x8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MARK RO 0x1	Reserved														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						P1 RW 0x2		Reserved						P0 RW 0x2	

fpga2soc_axi32_I_main_QosGenerator_Priority Fields

Bit	Name	Description	Access	Reset
31	MARK	Backward compatibility marker when 0.	RO	0x1

Bit	Name	Description	Access	Reset
9:8	P1	In Programmable or Bandwidth Limiter mode, the priority level for read transactions. In Bandwidth regulator mode, the priority level when the used throughput is below the threshold. In Bandwidth Regulator mode, P1 should have a value equal or greater than P0.	RW	0x2
1:0	P0	In Programmable or Bandwidth Limiter mode, the priority level for write transactions. In Bandwidth Regulator mode, the priority level when the used throughput is above the threshold. In Bandwidth Regulator mode, P0 should have a value equal or lower than P1.	RW	0x2

fpga2soc_axi32_I_main_QosGenerator_Mode

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2soc_axi32_I_main_QosGenerator	0xFFD16100	0xFFD1610C

Offset: 0xC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														MODE RW 0x1	

fpga2soc_axi32_I_main_QosGenerator_Mode Fields

Bit	Name	Description	Access	Reset
1:0	MODE	0 = Programmable mode: a programmed priority is assigned to each read or write, 1 = Bandwidth Limiter Mode: a hard limit restricts throughput, 2 = Bypass mode: (<See SoC-specific QoS generator documentation>), 3 = Bandwidth Regulator mode: priority decreases when throughput exceeds a threshold.	RW	0x1

fpga2soc_axi32_I_main_QosGenerator_Bandwidth

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2soc_axi32_I_main_QosGenerator	0xFFD16100	0xFFD16110

Offset: 0x10

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					BANDWIDTH RW 0x280										

fpga2soc_axi32_I_main_QosGenerator_Bandwidth Fields

Bit	Name	Description	Access	Reset
10:0	BANDWIDTH	In Bandwidth Limiter or Bandwidth Regulator mode, the bandwidth threshold is in units of 1/256th bytes per cycle. The formula to calculate this value is: Bandwidth = Throughput (Mbsp) * 256 / Clock Frequency. For example, 80 MBps on a 250 MHz interface has the value 0x0052 (BW = 80Mbps * 256/ 250MHz = 81.92, which is rounded to 0x0052).	RW	0x280

fpga2soc_axi32_I_main_QosGenerator_Saturation

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2soc_axi32_I_main_QosGenerator	0xFFD16100	0xFFD16114

Offset: 0x14

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						SATURATION RW 0x8									

fpga2soc_axi32_I_main_QosGenerator_Saturation Fields

Bit	Name	Description	Access	Reset
9:0	SATURATION	In Bandwidth Limiter or Bandwidth Regulator mode, the maximum data count value, in units of 16 bytes. This determines the window of time over which bandwidth is measured. For example, to measure bandwidth within a 1000 cycle window on a 64-bit interface is value 0x1F4.	RW	0x8

fpga2soc_axi32_I_main_QosGenerator_ExtControl

External inputs control.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2soc_axi32_I_main_QosGenerator	0xFFD16100	0xFFD16118

Offset: 0x18

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													INTCLKEN RW 0x0	EXTTHREN RW 0x0	SOCKETQOSEN RW 0x0

fpga2soc_axi32_I_main_QosGenerator_ExtControl Fields

Bit	Name	Description	Access	Reset
2	INTCLKEN	n/a	RW	0x0
1	EXTTHREN	n/a	RW	0x0

Bit	Name	Description	Access	Reset
0	SOCKETQOSEN	n/a	RW	0x0

noc_mpu_fpga2soc_axi64_I_main_QosGenerator Address Map

Module Instance	Base Address	End Address
i_noc_mpu_m0_fpga2soc_axi64_I_main_QosGenerator	0xFFD16180	0xFFD161FF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
fpga2soc_axi64_I_main_QosGenerator_Id_CoreId on page 7-692	0x0	32	RO	0x8256B104	
fpga2soc_axi64_I_main_QosGenerator_Id_RevisionId on page 7-693	0x4	32	RO	0x129FF00	
fpga2soc_axi64_I_main_QosGenerator_Priority on page 7-694	0x8	32	RW	0x80000202	Priority register.
fpga2soc_axi64_I_main_QosGenerator_Mode on page 7-695	0xC	32	RW	0x1	
fpga2soc_axi64_I_main_QosGenerator_Bandwidth on page 7-696	0x10	32	RW	0x780	
fpga2soc_axi64_I_main_QosGenerator_Saturation on page 7-697	0x14	32	RW	0x8	

Register	Offset	Width	Access	Reset Value	Description
fpga2soc_axi64_I_main_QosGenerator_ExtControl on page 7-698	0x18	32	RW	0x0	External inputs control.

noc_mpu_fpga2soc_axi64_I_main_QosGenerator Summary

Base Address: 0xFFD16180

Register Address Offset	Bit Fields															
i_noc_mpu_m0_fpga2soc_axi64_I_main_QosGenerator																
fpga2soc_axi64_I_main_QosGenerator_Id_CoreId 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	CORECHECKSUM RO 0x8256B1															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CORECHECKSUM RO 0x8256B1								CORETYPEID RO 0x4							
fpga2soc_axi64_I_main_QosGenerator_Id_RevisionId 0x4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	FLEXNOCID RO 0x129FF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FLEXNOCID RO 0x129FF								USERID RO 0x0							
fpga2soc_axi64_I_main_QosGenerator_Priority 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	MARK RO 0x1	Reserved														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						P1 RW 0x2		Reserved						P0 RW 0x2	

Register Address Offset	Bit Fields															
fpga2soc_axi64_I_main_QosGenerator_Mode 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														MODE RW 0x1	
fpga2soc_axi64_I_main_QosGenerator_Bandwidth 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				BANDWIDTH RW 0x780											
fpga2soc_axi64_I_main_QosGenerator_Saturation 0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						SATURATION RW 0x8									
fpga2soc_axi64_I_main_QosGenerator_ExtControl 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													INTCLKEN RW 0x0	EXTTHREN RW 0x0	SOCKETQOSEN RW 0x0

fpga2soc_axi64_I_main_QosGenerator_Id_CoreId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2soc_axi64_I_main_QosGenerator	0xFFD16180	0xFFD16180

Offset: 0x0

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CORECHECKSUM RO 0x8256B1															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CORECHECKSUM RO 0x8256B1								CORETYPEID RO 0x4							

fpga2soc_axi64_l_main_QosGenerator_Id_CoreId Fields

Bit	Name	Description	Access	Reset
31:8	CORECHECKSUM	Field containing a checksum of the parameters of the IP.	RO	0x8256B1
7:0	CORETYPEID	Field identifying the type of IP.	RO	0x4

fpga2soc_axi64_l_main_QosGenerator_Id_RevisionId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2soc_axi64_I_main_QosGenerator	0xFFD16180	0xFFD16184

Offset: 0x4

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLEXNOCID RO 0x129FF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLEXNOCID RO 0x129FF								USERID RO 0x0							

fpga2soc_axi64_I_main_QosGenerator_Id_RevisionId Fields

Bit	Name	Description	Access	Reset
31:8	FLEXNOCID	Field containing the build revision of the software used to generate the IP HDL code.	RO	0x129FF
7:0	USERID	Field containing a user defined value, not used anywhere inside the IP itself.	RO	0x0

fpga2soc_axi64_I_main_QosGenerator_Priority

Priority register.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2soc_axi64_I_main_QosGenerator	0xFFD16180	0xFFD16188

Offset: 0x8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MARK RO 0x1	Reserved														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						P1 RW 0x2		Reserved						P0 RW 0x2	

fpga2soc_axi64_I_main_QosGenerator_Priority Fields

Bit	Name	Description	Access	Reset
31	MARK	Backward compatibility marker when 0.	RO	0x1

Bit	Name	Description	Access	Reset
9:8	P1	In Programmable or Bandwidth Limiter mode, the priority level for read transactions. In Bandwidth regulator mode, the priority level when the used throughput is below the threshold. In Bandwidth Regulator mode, P1 should have a value equal or greater than P0.	RW	0x2
1:0	P0	In Programmable or Bandwidth Limiter mode, the priority level for write transactions. In Bandwidth Regulator mode, the priority level when the used throughput is above the threshold. In Bandwidth Regulator mode, P0 should have a value equal or lower than P1.	RW	0x2

fpga2soc_axi64_I_main_QosGenerator_Mode

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2soc_axi64_I_main_QosGenerator	0xFFD16180	0xFFD1618C

Offset: 0xC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														MODE RW 0x1	

fpga2soc_axi64_I_main_QosGenerator_Mode Fields

Bit	Name	Description	Access	Reset
1:0	MODE	0 = Programmable mode: a programmed priority is assigned to each read or write, 1 = Bandwidth Limiter Mode: a hard limit restricts throughput, 2 = Bypass mode: (<See SoC-specific QoS generator documentation>), 3 = Bandwidth Regulator mode: priority decreases when throughput exceeds a threshold.	RW	0x1

fpga2soc_axi64_I_main_QosGenerator_Bandwidth

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2soc_axi64_I_main_QosGenerator	0xFFD16180	0xFFD16190

Offset: 0x10

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				BANDWIDTH RW 0x780											

fpga2soc_axi64_I_main_QosGenerator_Bandwidth Fields

Bit	Name	Description	Access	Reset
11:0	BANDWIDTH	In Bandwidth Limiter or Bandwidth Regulator mode, the bandwidth threshold is in units of 1/256th bytes per cycle. The formula to calculate this value is: Bandwidth = Throughput (Mbsp) * 256 / Clock Frequency. For example, 80 MBps on a 250 MHz interface has the value 0x0052 (BW = 80Mbps * 256/ 250MHz = 81.92, which is rounded to 0x0052).	RW	0x780

fpga2soc_axi64_I_main_QosGenerator_Saturation

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2soc_axi64_I_main_QosGenerator	0xFFD16180	0xFFD16194

Offset: 0x14

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						SATURATION RW 0x8									

fpga2soc_axi64_l_main_QosGenerator_Saturation Fields

Bit	Name	Description	Access	Reset
9:0	SATURATION	In Bandwidth Limiter or Bandwidth Regulator mode, the maximum data count value, in units of 16 bytes. This determines the window of time over which bandwidth is measured. For example, to measure bandwidth within a 1000 cycle window on a 64-bit interface is value 0x1F4.	RW	0x8

fpga2soc_axi64_l_main_QosGenerator_ExtControl

External inputs control.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2soc_axi64_I_main_QosGenerator	0xFFD16180	0xFFD16198

Offset: 0x18

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													INTCLKEN RW 0x0	EXTTHREN RW 0x0	SOCKETQOSEN RW 0x0

fpga2soc_axi64_l_main_QosGenerator_ExtControl Fields

Bit	Name	Description	Access	Reset
2	INTCLKEN	n/a	RW	0x0
1	EXTTHREN	n/a	RW	0x0

Bit	Name	Description	Access	Reset
0	SOCKETQOSEN	n/a	RW	0x0

noc_mpu_fpga2soc_axi128_l_main_QosGenerator Address Map

Module Instance	Base Address	End Address
i_noc_mpu_m0_fpga2soc_axi128_l_main_QosGenerator	0xFFD16200	0xFFD1627F

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
fpga2soc_axi128_I_main_QosGenerator_Id_CoreId on page 7-701	0x0	32	RO	0xDC0D0004	
fpga2soc_axi128_I_main_QosGenerator_Id_RevisionId on page 7-702	0x4	32	RO	0x129FF00	
fpga2soc_axi128_I_main_QosGenerator_Priority on page 7-703	0x8	32	RW	0x80000202	Priority register.
fpga2soc_axi128_I_main_QosGenerator_Mode on page 7-704	0xC	32	RW	0x1	
fpga2soc_axi128_I_main_QosGenerator_Bandwidth on page 7-705	0x10	32	RW	0xC80	
fpga2soc_axi128_I_main_QosGenerator_Saturation on page 7-706	0x14	32	RW	0x8	

Register	Offset	Width	Access	Reset Value	Description
fpga2soc_axi128_I_main_QosGenerator_ExtControl on page 7-707	0x18	32	RW	0x0	External inputs control.

noc_mpu_fpga2soc_axi128_I_main_QosGenerator Summary

Base Address: 0xFFD16200

Register Address Offset	Bit Fields															
i_noc_mpu_m0_fpga2soc_axi128_I_main_QosGenerator																
fpga2soc_axi128_I_main_QosGenerator_Id_CoreId 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	CORECHECKSUM RO 0xDC0D00															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CORECHECKSUM RO 0xDC0D00								CORETYPEID RO 0x4							
fpga2soc_axi128_I_main_QosGenerator_Id_RevisionId 0x4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	FLEXNOCID RO 0x129FF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FLEXNOCID RO 0x129FF								USERID RO 0x0							
fpga2soc_axi128_I_main_QosGenerator_Priority 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	MARK RO 0x1	Reserved														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						P1 RW 0x2		Reserved						P0 RW 0x2	

Register Address Offset	Bit Fields															
fpga2soc_axi128_I_main_QosGenerator_Mode 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														MODE RW 0x1	
fpga2soc_axi128_I_main_QosGenerator_Bandwidth 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved			BANDWIDTH RW 0xC80												
fpga2soc_axi128_I_main_QosGenerator_Saturation 0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						SATURATION RW 0x8									
fpga2soc_axi128_I_main_QosGenerator_ExtControl 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													INTCLKEN RW 0x0	EXTTHREN RW 0x0	SOCKETQOSEN RW 0x0

fpga2soc_axi128_I_main_QosGenerator_Id_CoreId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2soc_axi128_I_main_QosGenerator	0xFFD16200	0xFFD16200

Offset: 0x0

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CORECHECKSUM RO 0xDC0D00															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CORECHECKSUM RO 0xDC0D00								CORETYPEID RO 0x4							

fpga2soc_axi128_I_main_QosGenerator_Id_CoreId Fields

Bit	Name	Description	Access	Reset
31:8	CORECHECKSUM	Field containing a checksum of the parameters of the IP.	RO	0xDC0D00
7:0	CORETYPEID	Field identifying the type of IP.	RO	0x4

fpga2soc_axi128_I_main_QosGenerator_Id_RevisionId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2soc_axi128_I_main_QosGenerator	0xFFD16200	0xFFD16204

Offset: 0x4

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLEXNOCID RO 0x129FF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLEXNOCID RO 0x129FF								USERID RO 0x0							

fpga2soc_axi128_l_main_QosGenerator_Id_RevisionId Fields

Bit	Name	Description	Access	Reset
31:8	FLEXNOCID	Field containing the build revision of the software used to generate the IP HDL code.	RO	0x129FF
7:0	USERID	Field containing a user defined value, not used anywhere inside the IP itself.	RO	0x0

fpga2soc_axi128_l_main_QosGenerator_Priority

Priority register.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2soc_axi128_I_main_QosGenerator	0xFFD16200	0xFFD16208

Offset: 0x8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MARK RO 0x1	Reserved														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						P1 RW 0x2		Reserved						P0 RW 0x2	

fpga2soc_axi128_l_main_QosGenerator_Priority Fields

Bit	Name	Description	Access	Reset
31	MARK	Backward compatibility marker when 0.	RO	0x1

Bit	Name	Description	Access	Reset
9:8	P1	In Programmable or Bandwidth Limiter mode, the priority level for read transactions. In Bandwidth regulator mode, the priority level when the used throughput is below the threshold. In Bandwidth Regulator mode, P1 should have a value equal or greater than P0.	RW	0x2
1:0	P0	In Programmable or Bandwidth Limiter mode, the priority level for write transactions. In Bandwidth Regulator mode, the priority level when the used throughput is above the threshold. In Bandwidth Regulator mode, P0 should have a value equal or lower than P1.	RW	0x2

fpga2soc_axi128_I_main_QosGenerator_Mode

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2soc_axi128_I_main_QosGenerator	0xFFD16200	0xFFD1620C

Offset: 0xC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														MODE RW 0x1	

fpga2soc_axi128_I_main_QosGenerator_Mode Fields

Bit	Name	Description	Access	Reset
1:0	MODE	0 = Programmable mode: a programmed priority is assigned to each read or write, 1 = Bandwidth Limiter Mode: a hard limit restricts throughput, 2 = Bypass mode: (<See SoC-specific QoS generator documentation>), 3 = Bandwidth Regulator mode: priority decreases when throughput exceeds a threshold.	RW	0x1

fpga2soc_axi128_I_main_QosGenerator_Bandwidth

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2soc_axi128_I_main_QosGenerator	0xFFD16200	0xFFD16210

Offset: 0x10

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			BANDWIDTH RW 0xC80												

fpga2soc_axi128_l_main_QosGenerator_Bandwidth Fields

Bit	Name	Description	Access	Reset
12:0	BANDWIDTH	In Bandwidth Limiter or Bandwidth Regulator mode, the bandwidth threshold is in units of 1/256th bytes per cycle. The formula to calculate this value is: Bandwidth = Throughput (Mbsp) * 256 / Clock Frequency. For example, 80 MBps on a 250 MHz interface has the value 0x0052 (BW = 80Mbps * 256/ 250MHz = 81.92, which is rounded to 0x0052).	RW	0xC80

fpga2soc_axi128_l_main_QosGenerator_Saturation

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2soc_axi128_l_main_QosGenerator	0xFFD16200	0xFFD16214

Offset: 0x14

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						SATURATION RW 0x8									

fpga2soc_axi128_I_main_QosGenerator_Saturation Fields

Bit	Name	Description	Access	Reset
9:0	SATURATION	In Bandwidth Limiter or Bandwidth Regulator mode, the maximum data count value, in units of 16 bytes. This determines the window of time over which bandwidth is measured. For example, to measure bandwidth within a 1000 cycle window on a 64-bit interface is value 0x1F4.	RW	0x8

fpga2soc_axi128_I_main_QosGenerator_ExtControl

External inputs control.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2soc_axi128_I_main_QosGenerator	0xFFD16200	0xFFD16218

Offset: 0x18

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													INTCLKEN RW 0x0	EXTTHREN RW 0x0	SOCKETQOSEN RW 0x0

fpga2soc_axi128_I_main_QosGenerator_ExtControl Fields

Bit	Name	Description	Access	Reset
2	INTCLKEN	n/a	RW	0x0
1	EXTTHREN	n/a	RW	0x0

Bit	Name	Description	Access	Reset
0	SOCKETQOSEN	n/a	RW	0x0

noc_mpu_dma_I_main_QosGenerator Address Map

Module Instance	Base Address	End Address
i_noc_mpu_m0_dma_m0_I_main_QosGenerator	0xFFD16280	0xFFD162FF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
dma_m0_I_main_QosGenerator_Id_CoreId on page 7-710	0x0	32	RO	0xAEE31604	
dma_m0_I_main_QosGenerator_Id_RevisionId on page 7-711	0x4	32	RO	0x129FF00	
dma_m0_I_main_QosGenerator_Priority on page 7-712	0x8	32	RW	0x80000101	Priority register.
dma_m0_I_main_QosGenerator_Mode on page 7-713	0xC	32	RW	0x1	
dma_m0_I_main_QosGenerator_Bandwidth on page 7-714	0x10	32	RW	0x280	
dma_m0_I_main_QosGenerator_Saturation on page 7-715	0x14	32	RW	0x4	
dma_m0_I_main_QosGenerator_ExtControl on page 7-716	0x18	32	RW	0x0	External inputs control.

noc_mpu_dma_I_main_QosGenerator Summary

Base Address: 0xFFD16280

Register Address Offset	Bit Fields															
i_noc_mpu_m0_dma_m0_I_main_QosGener-ator																
dma_m0_I_main_QosGenerator_Id_CoreId 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	CORECHECKSUM RO 0xAEE316															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CORECHECKSUM RO 0xAEE316								CORETYPEID RO 0x4							
dma_m0_I_main_QosGenerator_Id_RevisionId 0x4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	FLEXNOCID RO 0x129FF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FLEXNOCID RO 0x129FF								USERID RO 0x0							
dma_m0_I_main_QosGenerator_Priority 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	MARK RO 0x1	Reserved														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						P1 RW 0x1		Reserved						P0 RW 0x1	
dma_m0_I_main_QosGenerator_Mode 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														MODE RW 0x1	
dma_m0_I_main_QosGenerator_Bandwidth 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				BANDWIDTH RW 0x280											

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dma_m0_I_main_QosGenerator_Saturation 0x14	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						SATURATION RW 0x4									
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dma_m0_I_main_QosGenerator_ExtControl 0x18	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													INTCLKEN RW 0x0	EXTTHREN RW 0x0	SOCKETQOSEN RW 0x0

dma_m0_I_main_QosGenerator_Id_CoreId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_dma_m0_I_main_QosGenerator	0xFFD16280	0xFFD16280

Offset: 0x0

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CORECHECKSUM RO 0xAEE316															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CORECHECKSUM RO 0xAEE316								CORETYPEID RO 0x4							

dma_m0_I_main_QosGenerator_Id_CoreId Fields

Bit	Name	Description	Access	Reset
31:8	CORECHECKSUM	Field containing a checksum of the parameters of the IP.	RO	0xAEE316
7:0	CORETYPEID	Field identifying the type of IP.	RO	0x4

dma_m0_I_main_QosGenerator_Id_RevisionId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_dma_m0_I_main_QosGenerator	0xFFD16280	0xFFD16284

Offset: 0x4

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLEXNOCID RO 0x129FF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLEXNOCID RO 0x129FF								USERID RO 0x0							

dma_m0_I_main_QosGenerator_Id_RevisionId Fields

Bit	Name	Description	Access	Reset
31:8	FLEXNOCID	Field containing the build revision of the software used to generate the IP HDL code.	RO	0x129FF
7:0	USERID	Field containing a user defined value, not used anywhere inside the IP itself.	RO	0x0

dma_m0_I_main_QosGenerator_Priority

Priority register.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_dma_m0_I_main_QosGenerator	0xFFD16280	0xFFD16288

Offset: 0x8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MARK RO 0x1	Reserved														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						P1 RW 0x1		Reserved						P0 RW 0x1	

dma_m0_I_main_QosGenerator_Priority Fields

Bit	Name	Description	Access	Reset
31	MARK	Backward compatibility marker when 0.	RO	0x1
9:8	P1	In Programmable or Bandwidth Limiter mode, the priority level for read transactions. In Bandwidth regulator mode, the priority level when the used throughput is below the threshold. In Bandwidth Regulator mode, P1 should have a value equal or greater than P0.	RW	0x1

Bit	Name	Description	Access	Reset
1:0	P0	In Programmable or Bandwidth Limiter mode, the priority level for write transactions. In Bandwidth Regulator mode, the priority level when the used throughput is above the threshold. In Bandwidth Regulator mode, P0 should have a value equal or lower than P1.	RW	0x1

dma_m0_I_main_QosGenerator_Mode

Module Instance	Base Address	Register Address
i_noc_mpu_m0_dma_m0_I_main_QosGenerator	0xFFD16280	0xFFD1628C

Offset: 0xC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														MODE RW 0x1	

dma_m0_I_main_QosGenerator_Mode Fields

Bit	Name	Description	Access	Reset
1:0	MODE	0 = Programmable mode: a programmed priority is assigned to each read or write, 1 = Bandwidth Limiter Mode: a hard limit restricts throughput, 2 = Bypass mode: (<See SoC-specific QoS generator documentation>), 3 = Bandwidth Regulator mode: priority decreases when throughput exceeds a threshold.	RW	0x1

dma_m0_I_main_QosGenerator_Bandwidth

Module Instance	Base Address	Register Address
i_noc_mpu_m0_dma_m0_I_main_QosGenerator	0xFFD16280	0xFFD16290

Offset: 0x10

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				BANDWIDTH RW 0x280											

dma_m0_I_main_QosGenerator_Bandwidth Fields

Bit	Name	Description	Access	Reset
11:0	BANDWIDTH	In Bandwidth Limiter or Bandwidth Regulator mode, the bandwidth threshold is in units of 1/256th bytes per cycle. The formula to calculate this value is: Bandwidth = Throughput (Mbsp) * 256 / Clock Frequency. For example, 80 MBps on a 250 MHz interface has the value 0x0052 (BW = 80Mbps * 256/ 250MHz = 81.92, which is rounded to 0x0052).	RW	0x280

dma_m0_I_main_QosGenerator_Saturation

Module Instance	Base Address	Register Address
i_noc_mpu_m0_dma_m0_I_main_QosGenerator	0xFFD16280	0xFFD16294

Offset: 0x14

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						SATURATION RW 0x4									

dma_m0_I_main_QosGenerator_Saturation Fields

Bit	Name	Description	Access	Reset
9:0	SATURATION	In Bandwidth Limiter or Bandwidth Regulator mode, the maximum data count value, in units of 16 bytes. This determines the window of time over which bandwidth is measured. For example, to measure bandwidth within a 1000 cycle window on a 64-bit interface is value 0x1F4.	RW	0x4

dma_m0_I_main_QosGenerator_ExtControl

External inputs control.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_dma_m0_I_main_QosGenerator	0xFFD16280	0xFFD16298

Offset: 0x18

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													INTCLKEN RW 0x0	EXTTHREN RW 0x0	SOCKETQOSEN RW 0x0

dma_m0_I_main_QosGenerator_ExtControl Fields

Bit	Name	Description	Access	Reset
2	INTCLKEN	n/a	RW	0x0
1	EXTTHREN	n/a	RW	0x0

Bit	Name	Description	Access	Reset
0	SOCKETQOSEN	n/a	RW	0x0

noc_mpu_emac0_m_I_main_QosGenerator Address Map

Module Instance	Base Address	End Address
i_noc_mpu_m0_emac0_m_I_main_QosGenerator	0xFFD16300	0xFFD1637F

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
emac0_m_I_main_QosGenerator_Id_CoreId on page 7-719	0x0	32	RO	0x7D482104	
emac0_m_I_main_QosGenerator_Id_RevisionId on page 7-720	0x4	32	RO	0x129FF00	
emac0_m_I_main_QosGenerator_Priority on page 7-721	0x8	32	RW	0x80000100	Priority register.
emac0_m_I_main_QosGenerator_Mode on page 7-722	0xC	32	RW	0x3	
emac0_m_I_main_QosGenerator_Bandwidth on page 7-723	0x10	32	RW	0x100	
emac0_m_I_main_QosGenerator_Saturation on page 7-724	0x14	32	RW	0x4	
emac0_m_I_main_QosGenerator_ExtControl on page 7-725	0x18	32	RW	0x0	External inputs control.

noc_mpu_emac0_m_I_main_QosGenerator Summary

Base Address: 0xFFD16300

Register Address Offset	Bit Fields															
i_noc_mpu_m0_emac0_m_I_main_QosGenerator																
emac0_m_I_main_QosGenerator_Id_CoreId 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	CORECHECKSUM RO 0x7D4821															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CORECHECKSUM RO 0x7D4821								CORETYPEID RO 0x4							
emac0_m_I_main_QosGenerator_Id_RevisionId 0x4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	FLEXNOCID RO 0x129FF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FLEXNOCID RO 0x129FF								USERID RO 0x0							
emac0_m_I_main_QosGenerator_Priority 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	MARK RO 0x1	Reserved														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						P1 RW 0x1		Reserved						P0 RW 0x0	
emac0_m_I_main_QosGenerator_Mode 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														MODE RW 0x3	
emac0_m_I_main_QosGenerator_Bandwidth 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved					BANDWIDTH RW 0x100										

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
emac0_m_I_main_QosGenerator_Saturation 0x14	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						SATURATION RW 0x4									
emac0_m_I_main_QosGenerator_ExtControl 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													INTCLKEN RW 0x0	EXTTHREN RW 0x0	SOCKETQOSEN RW 0x0

emac0_m_I_main_QosGenerator_Id_CoreId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_emac0_m_I_main_QosGenerator	0xFFD16300	0xFFD16300

Offset: 0x0

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CORECHECKSUM RO 0x7D4821															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CORECHECKSUM RO 0x7D4821								CORETYPEID RO 0x4							

emac0_m_I_main_QosGenerator_Id_CoreId Fields

Bit	Name	Description	Access	Reset
31:8	CORECHECKSUM	Field containing a checksum of the parameters of the IP.	RO	0x7D4821
7:0	CORETYPEID	Field identifying the type of IP.	RO	0x4

emac0_m_I_main_QosGenerator_Id_RevisionId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_emac0_m_I_main_QosGenerator	0xFFD16300	0xFFD16304

Offset: 0x4

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLEXNOCID RO 0x129FF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLEXNOCID RO 0x129FF								USERID RO 0x0							

emac0_m_I_main_QosGenerator_Id_RevisionId Fields

Bit	Name	Description	Access	Reset
31:8	FLEXNOCID	Field containing the build revision of the software used to generate the IP HDL code.	RO	0x129FF
7:0	USERID	Field containing a user defined value, not used anywhere inside the IP itself.	RO	0x0

emac0_m_I_main_QosGenerator_Priority

Priority register.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_emac0_m_I_main_QosGenerator	0xFFD16300	0xFFD16308

Offset: 0x8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MARK RO 0x1	Reserved														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						P1 RW 0x1		Reserved						P0 RW 0x0	

emac0_m_I_main_QosGenerator_Priority Fields

Bit	Name	Description	Access	Reset
31	MARK	Backward compatibility marker when 0.	RO	0x1
9:8	P1	In Programmable or Bandwidth Limiter mode, the priority level for read transactions. In Bandwidth regulator mode, the priority level when the used throughput is below the threshold. In Bandwidth Regulator mode, P1 should have a value equal or greater than P0.	RW	0x1

Bit	Name	Description	Access	Reset
1:0	P0	In Programmable or Bandwidth Limiter mode, the priority level for write transactions. In Bandwidth Regulator mode, the priority level when the used throughput is above the threshold. In Bandwidth Regulator mode, P0 should have a value equal or lower than P1.	RW	0x0

emac0_m_I_main_QosGenerator_Mode

Module Instance	Base Address	Register Address
i_noc_mpu_m0_emac0_m_I_main_QosGenerator	0xFFD16300	0xFFD1630C

Offset: 0xC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														MODE RW 0x3	

emac0_m_l_main_QosGenerator_Mode Fields

Bit	Name	Description	Access	Reset
1:0	MODE	0 = Programmable mode: a programmed priority is assigned to each read or write, 1 = Bandwidth Limiter Mode: a hard limit restricts throughput, 2 = Bypass mode: (<See SoC-specific QoS generator documentation>), 3 = Bandwidth Regulator mode: priority decreases when throughput exceeds a threshold.	RW	0x3

emac0_m_l_main_QosGenerator_Bandwidth

Module Instance	Base Address	Register Address
i_noc_mpu_m0_emac0_m_I_main_QosGenerator	0xFFD16300	0xFFD16310

Offset: 0x10

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					BANDWIDTH RW 0x100										

emac0_m_I_main_QosGenerator_Bandwidth Fields

Bit	Name	Description	Access	Reset
10:0	BANDWIDTH	In Bandwidth Limiter or Bandwidth Regulator mode, the bandwidth threshold is in units of 1/256th bytes per cycle. The formula to calculate this value is: Bandwidth = Throughput (Mbsp) * 256 / Clock Frequency. For example, 80 MBps on a 250 MHz interface has the value 0x0052 (BW = 80Mbps * 256/ 250MHz = 81.92, which is rounded to 0x0052).	RW	0x100

emac0_m_I_main_QosGenerator_Saturation

Module Instance	Base Address	Register Address
i_noc_mpu_m0_emac0_m_I_main_QosGenerator	0xFFD16300	0xFFD16314

Offset: 0x14

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						SATURATION RW 0x4									

emac0_m_l_main_QosGenerator_Saturation Fields

Bit	Name	Description	Access	Reset
9:0	SATURATION	In Bandwidth Limiter or Bandwidth Regulator mode, the maximum data count value, in units of 16 bytes. This determines the window of time over which bandwidth is measured. For example, to measure bandwidth within a 1000 cycle window on a 64-bit interface is value 0x1F4.	RW	0x4

emac0_m_l_main_QosGenerator_ExtControl

External inputs control.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_emac0_m_I_main_QosGenerator	0xFFD16300	0xFFD16318

Offset: 0x18

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													INTCLKEN RW 0x0	EXTTHREN RW 0x0	SOCKETQOSEN RW 0x0

emac0_m_l_main_QosGenerator_ExtControl Fields

Bit	Name	Description	Access	Reset
2	INTCLKEN	n/a	RW	0x0
1	EXTTHREN	n/a	RW	0x0

Bit	Name	Description	Access	Reset
0	SOCKETQOSEN	n/a	RW	0x0

noc_mpu_emac1_m_I_main_QosGenerator Address Map

Module Instance	Base Address	End Address
i_noc_mpu_m0_emac1_m_I_main_QosGenerator	0xFFD16380	0xFFD163FF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
emac1_m_I_main_QosGenerator_Id_CoreId on page 7-728	0x0	32	RO	0x7844AD04	
emac1_m_I_main_QosGenerator_Id_RevisionId on page 7-729	0x4	32	RO	0x129FF00	
emac1_m_I_main_QosGenerator_Priority on page 7-730	0x8	32	RW	0x80000100	Priority register.
emac1_m_I_main_QosGenerator_Mode on page 7-731	0xC	32	RW	0x3	
emac1_m_I_main_QosGenerator_Bandwidth on page 7-732	0x10	32	RW	0x100	
emac1_m_I_main_QosGenerator_Saturation on page 7-733	0x14	32	RW	0x4	
emac1_m_I_main_QosGenerator_ExtControl on page 7-734	0x18	32	RW	0x0	External inputs control.

noc_mpu_emac1_m_I_main_QosGenerator Summary

Base Address: 0xFFD16380

Register Address Offset	Bit Fields															
i_noc_mpu_m0_emac1_m_I_main_QosGener-ator																
emac1_m_I_main_QosGenerator_Id_CoreId 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	CORECHECKSUM RO 0x7844AD															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CORECHECKSUM RO 0x7844AD								CORETYPEID RO 0x4							
emac1_m_I_main_QosGenerator_Id_RevisionId 0x4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	FLEXNOCID RO 0x129FF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FLEXNOCID RO 0x129FF								USERID RO 0x0							
emac1_m_I_main_QosGenerator_Priority 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	MARK RO 0x1	Reserved														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						P1 RW 0x1		Reserved						P0 RW 0x0	
emac1_m_I_main_QosGenerator_Mode 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														MODE RW 0x3	
emac1_m_I_main_QosGenerator_Bandwidth 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved					BANDWIDTH RW 0x100										

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
emac1_m_I_main_QosGenerator_Saturation 0x14	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						SATURATION RW 0x4									
emac1_m_I_main_QosGenerator_ExtControl 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													INTCLKEN RW 0x0	EXTTHREN RW 0x0	SOCKETQOSEN RW 0x0

emac1_m_I_main_QosGenerator_Id_CoreId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_emac1_m_I_main_QosGenerator	0xFFD16380	0xFFD16380

Offset: 0x0

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CORECHECKSUM RO 0x7844AD															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CORECHECKSUM RO 0x7844AD								CORETYPEID RO 0x4							

emac1_m_I_main_QosGenerator_Id_CoreId Fields

Bit	Name	Description	Access	Reset
31:8	CORECHECKSUM	Field containing a checksum of the parameters of the IP.	RO	0x7844AD
7:0	CORETYPEID	Field identifying the type of IP.	RO	0x4

emac1_m_I_main_QosGenerator_Id_RevisionId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_emac1_m_I_main_QosGenerator	0xFFD16380	0xFFD16384

Offset: 0x4

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLEXNOCID RO 0x129FF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLEXNOCID RO 0x129FF								USERID RO 0x0							

emac1_m_I_main_QosGenerator_Id_RevisionId Fields

Bit	Name	Description	Access	Reset
31:8	FLEXNOCID	Field containing the build revision of the software used to generate the IP HDL code.	RO	0x129FF
7:0	USERID	Field containing a user defined value, not used anywhere inside the IP itself.	RO	0x0

emac1_m_I_main_QosGenerator_Priority

Priority register.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_emac1_m_I_main_QosGenerator	0xFFD16380	0xFFD16388

Offset: 0x8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MARK RO 0x1	Reserved														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						P1 RW 0x1		Reserved						P0 RW 0x0	

emac1_m_I_main_QosGenerator_Priority Fields

Bit	Name	Description	Access	Reset
31	MARK	Backward compatibility marker when 0.	RO	0x1
9:8	P1	In Programmable or Bandwidth Limiter mode, the priority level for read transactions. In Bandwidth regulator mode, the priority level when the used throughput is below the threshold. In Bandwidth Regulator mode, P1 should have a value equal or greater than P0.	RW	0x1

Bit	Name	Description	Access	Reset
1:0	P0	In Programmable or Bandwidth Limiter mode, the priority level for write transactions. In Bandwidth Regulator mode, the priority level when the used throughput is above the threshold. In Bandwidth Regulator mode, P0 should have a value equal or lower than P1.	RW	0x0

emac1_m_I_main_QosGenerator_Mode

Module Instance	Base Address	Register Address
i_noc_mpu_m0_emac1_m_I_main_QosGenerator	0xFFD16380	0xFFD1638C

Offset: 0xC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														MODE RW 0x3	

emac1_m_I_main_QosGenerator_Mode Fields

Bit	Name	Description	Access	Reset
1:0	MODE	0 = Programmable mode: a programmed priority is assigned to each read or write, 1 = Bandwidth Limiter Mode: a hard limit restricts throughput, 2 = Bypass mode: (<See SoC-specific QoS generator documentation>), 3 = Bandwidth Regulator mode: priority decreases when throughput exceeds a threshold.	RW	0x3

emac1_m_I_main_QosGenerator_Bandwidth

Module Instance	Base Address	Register Address
i_noc_mpu_m0_emac1_m_I_main_QosGenerator	0xFFD16380	0xFFD16390

Offset: 0x10

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					BANDWIDTH RW 0x100										

emac1_m_I_main_QosGenerator_Bandwidth Fields

Bit	Name	Description	Access	Reset
10:0	BANDWIDTH	In Bandwidth Limiter or Bandwidth Regulator mode, the bandwidth threshold is in units of 1/256th bytes per cycle. The formula to calculate this value is: Bandwidth = Throughput (Mbsp) * 256 / Clock Frequency. For example, 80 MBps on a 250 MHz interface has the value 0x0052 (BW = 80Mbps * 256/ 250MHz = 81.92, which is rounded to 0x0052).	RW	0x100

emac1_m_I_main_QosGenerator_Saturation

Module Instance	Base Address	Register Address
i_noc_mpu_m0_emac1_m_I_main_QosGenerator	0xFFD16380	0xFFD16394

Offset: 0x14

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						SATURATION RW 0x4									

emac1_m_I_main_QosGenerator_Saturation Fields

Bit	Name	Description	Access	Reset
9:0	SATURATION	In Bandwidth Limiter or Bandwidth Regulator mode, the maximum data count value, in units of 16 bytes. This determines the window of time over which bandwidth is measured. For example, to measure bandwidth within a 1000 cycle window on a 64-bit interface is value 0x1F4.	RW	0x4

emac1_m_I_main_QosGenerator_ExtControl

External inputs control.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_emac1_m_I_main_QosGenerator	0xFFD16380	0xFFD16398

Offset: 0x18

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													INTCLKEN RW 0x0	EXTTHREN RW 0x0	SOCKETQOSEN RW 0x0

emac1_m_I_main_QosGenerator_ExtControl Fields

Bit	Name	Description	Access	Reset
2	INTCLKEN	n/a	RW	0x0
1	EXTTHREN	n/a	RW	0x0

Bit	Name	Description	Access	Reset
0	SOCKETQOSEN	n/a	RW	0x0

noc_mpu_emac2_m_I_main_QosGenerator Address Map

Module Instance	Base Address	End Address
i_noc_mpu_m0_emac2_m_I_main_QosGenerator	0xFFD16400	0xFFD1647F

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
emac2_m_I_main_QosGenerator_Id_CoreId on page 7-737	0x0	32	RO	0x81DC9B04	
emac2_m_I_main_QosGenerator_Id_RevisionId on page 7-738	0x4	32	RO	0x129FF00	
emac2_m_I_main_QosGenerator_Priority on page 7-739	0x8	32	RW	0x80000100	Priority register.
emac2_m_I_main_QosGenerator_Mode on page 7-740	0xC	32	RW	0x3	
emac2_m_I_main_QosGenerator_Bandwidth on page 7-741	0x10	32	RW	0x100	
emac2_m_I_main_QosGenerator_Saturation on page 7-742	0x14	32	RW	0x4	
emac2_m_I_main_QosGenerator_ExtControl on page 7-743	0x18	32	RW	0x0	External inputs control.

noc_mpu_emac2_m_I_main_QosGenerator Summary

Base Address: 0xFFD16400

Register Address Offset	Bit Fields															
i_noc_mpu_m0_emac2_m_I_main_QosGenerator																
emac2_m_I_main_QosGenerator_Id_CoreId 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	CORECHECKSUM RO 0x81DC9B															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CORECHECKSUM RO 0x81DC9B								CORETYPEID RO 0x4							
emac2_m_I_main_QosGenerator_Id_RevisionId 0x4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	FLEXNOCID RO 0x129FF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FLEXNOCID RO 0x129FF								USERID RO 0x0							
emac2_m_I_main_QosGenerator_Priority 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	MARK RO 0x1	Reserved														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						P1 RW 0x1		Reserved						P0 RW 0x0	
emac2_m_I_main_QosGenerator_Mode 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														MODE RW 0x3	
emac2_m_I_main_QosGenerator_Bandwidth 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved					BANDWIDTH RW 0x100										

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
emac2_m_I_main_QosGenerator_Saturation 0x14	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						SATURATION RW 0x4									
emac2_m_I_main_QosGenerator_ExtControl 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													INTCLKEN RW 0x0	EXTTHREN RW 0x0	SOCKETQOSEN RW 0x0

emac2_m_I_main_QosGenerator_Id_CoreId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_emac2_m_I_main_QosGenerator	0xFFD16400	0xFFD16400

Offset: 0x0

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CORECHECKSUM RO 0x81DC9B															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CORECHECKSUM RO 0x81DC9B								CORETYPEID RO 0x4							

emac2_m_I_main_QosGenerator_Id_CoreId Fields

Bit	Name	Description	Access	Reset
31:8	CORECHECKSUM	Field containing a checksum of the parameters of the IP.	RO	0x81DC9B
7:0	CORETYPEID	Field identifying the type of IP.	RO	0x4

emac2_m_I_main_QosGenerator_Id_RevisionId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_emac2_m_I_main_QosGenerator	0xFFD16400	0xFFD16404

Offset: 0x4

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLEXNOCID RO 0x129FF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLEXNOCID RO 0x129FF								USERID RO 0x0							

emac2_m_I_main_QosGenerator_Id_RevisionId Fields

Bit	Name	Description	Access	Reset
31:8	FLEXNOCID	Field containing the build revision of the software used to generate the IP HDL code.	RO	0x129FF
7:0	USERID	Field containing a user defined value, not used anywhere inside the IP itself.	RO	0x0

emac2_m_I_main_QosGenerator_Priority

Priority register.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_emac2_m_I_main_QosGenerator	0xFFD16400	0xFFD16408

Offset: 0x8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MARK RO 0x1	Reserved														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						P1 RW 0x1		Reserved						P0 RW 0x0	

emac2_m_I_main_QosGenerator_Priority Fields

Bit	Name	Description	Access	Reset
31	MARK	Backward compatibility marker when 0.	RO	0x1
9:8	P1	In Programmable or Bandwidth Limiter mode, the priority level for read transactions. In Bandwidth regulator mode, the priority level when the used throughput is below the threshold. In Bandwidth Regulator mode, P1 should have a value equal or greater than P0.	RW	0x1

Bit	Name	Description	Access	Reset
1:0	P0	In Programmable or Bandwidth Limiter mode, the priority level for write transactions. In Bandwidth Regulator mode, the priority level when the used throughput is above the threshold. In Bandwidth Regulator mode, P0 should have a value equal or lower than P1.	RW	0x0

emac2_m_I_main_QosGenerator_Mode

Module Instance	Base Address	Register Address
i_noc_mpu_m0_emac2_m_I_main_QosGenerator	0xFFD16400	0xFFD1640C

Offset: 0xC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														MODE RW 0x3	

emac2_m_I_main_QosGenerator_Mode Fields

Bit	Name	Description	Access	Reset
1:0	MODE	0 = Programmable mode: a programmed priority is assigned to each read or write, 1 = Bandwidth Limiter Mode: a hard limit restricts throughput, 2 = Bypass mode: (<See SoC-specific QoS generator documentation>), 3 = Bandwidth Regulator mode: priority decreases when throughput exceeds a threshold.	RW	0x3

emac2_m_I_main_QosGenerator_Bandwidth

Module Instance	Base Address	Register Address
i_noc_mpu_m0_emac2_m_I_main_QosGenerator	0xFFD16400	0xFFD16410

Offset: 0x10

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					BANDWIDTH RW 0x100										

emac2_m_I_main_QosGenerator_Bandwidth Fields

Bit	Name	Description	Access	Reset
10:0	BANDWIDTH	In Bandwidth Limiter or Bandwidth Regulator mode, the bandwidth threshold is in units of 1/256th bytes per cycle. The formula to calculate this value is: Bandwidth = Throughput (Mbsp) * 256 / Clock Frequency. For example, 80 MBps on a 250 MHz interface has the value 0x0052 (BW = 80Mbps * 256/ 250MHz = 81.92, which is rounded to 0x0052).	RW	0x100

emac2_m_I_main_QosGenerator_Saturation

Module Instance	Base Address	Register Address
i_noc_mpu_m0_emac2_m_I_main_QosGenerator	0xFFD16400	0xFFD16414

Offset: 0x14

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						SATURATION RW 0x4									

emac2_m_l_main_QosGenerator_Saturation Fields

Bit	Name	Description	Access	Reset
9:0	SATURATION	In Bandwidth Limiter or Bandwidth Regulator mode, the maximum data count value, in units of 16 bytes. This determines the window of time over which bandwidth is measured. For example, to measure bandwidth within a 1000 cycle window on a 64-bit interface is value 0x1F4.	RW	0x4

emac2_m_l_main_QosGenerator_ExtControl

External inputs control.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_emac2_m_I_main_QosGenerator	0xFFD16400	0xFFD16418

Offset: 0x18

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													INTCLKEN RW 0x0	EXTTHREN RW 0x0	SOCKETQOSEN RW 0x0

emac2_m_l_main_QosGenerator_ExtControl Fields

Bit	Name	Description	Access	Reset
2	INTCLKEN	n/a	RW	0x0
1	EXTTHREN	n/a	RW	0x0

Bit	Name	Description	Access	Reset
0	SOCKETQOSEN	n/a	RW	0x0

noc_mpu_usb0_m_I_main_QosGenerator Address Map

Module Instance	Base Address	End Address
i_noc_mpu_m0_usb0_m_I_main_QosGenerator	0xFFD16480	0xFFD164FF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
usb0_m_I_main_QosGenerator_Id_CoreId on page 7-746	0x0	32	RO	0x69BBC204	
usb0_m_I_main_QosGenerator_Id_RevisionId on page 7-747	0x4	32	RO	0x129FF00	
usb0_m_I_main_QosGenerator_Priority on page 7-748	0x8	32	RW	0x80000100	Priority register.
usb0_m_I_main_QosGenerator_Mode on page 7-749	0xC	32	RW	0x3	
usb0_m_I_main_QosGenerator_Bandwidth on page 7-750	0x10	32	RW	0x100	
usb0_m_I_main_QosGenerator_Saturation on page 7-751	0x14	32	RW	0x4	
usb0_m_I_main_QosGenerator_ExtControl on page 7-752	0x18	32	RW	0x0	External inputs control.

noc_mpu_usb0_m_I_main_QosGenerator Summary

Base Address: 0xFFD16480

Register Address Offset	Bit Fields															
i_noc_mpu_m0_usb0_m_I_main_QosGener-ator																
usb0_m_I_main_QosGenerator_Id_CoreId 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	CORECHECKSUM RO 0x69BBC2															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CORECHECKSUM RO 0x69BBC2								CORETYPEID RO 0x4							
usb0_m_I_main_QosGenerator_Id_RevisionId 0x4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	FLEXNOCID RO 0x129FF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FLEXNOCID RO 0x129FF								USERID RO 0x0							
usb0_m_I_main_QosGenerator_Priority 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	MARK RO 0x1	Reserved														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						P1 RW 0x1		Reserved						P0 RW 0x0	
usb0_m_I_main_QosGenerator_Mode 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														MODE RW 0x3	
usb0_m_I_main_QosGenerator_Bandwidth 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved					BANDWIDTH RW 0x100										

Register Address Offset	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
usb0_m_I_main_QosGenerator_Saturation 0x14	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						SATURATION RW 0x4									
usb0_m_I_main_QosGenerator_ExtControl 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													INTCLKEN RW 0x0	EXTTHREN RW 0x0	SOCKETQOSEN RW 0x0

usb0_m_I_main_QosGenerator_Id_CoreId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_usb0_m_I_main_QosGenerator	0xFFD16480	0xFFD16480

Offset: 0x0

Access: RO

Bit Fields																
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CORECHECKSUM RO 0x69BBC2															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CORECHECKSUM RO 0x69BBC2								CORETYPEID RO 0x4							

usb0_m_I_main_QosGenerator_Id_CoreId Fields

Bit	Name	Description	Access	Reset
31:8	CORECHECKSUM	Field containing a checksum of the parameters of the IP.	RO	0x69BBC2
7:0	CORETYPEID	Field identifying the type of IP.	RO	0x4

usb0_m_I_main_QosGenerator_Id_RevisionId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_usb0_m_I_main_QosGenerator	0xFFD16480	0xFFD16484

Offset: 0x4

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLEXNOCID RO 0x129FF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLEXNOCID RO 0x129FF								USERID RO 0x0							

usb0_m_I_main_QosGenerator_Id_RevisionId Fields

Bit	Name	Description	Access	Reset
31:8	FLEXNOCID	Field containing the build revision of the software used to generate the IP HDL code.	RO	0x129FF
7:0	USERID	Field containing a user defined value, not used anywhere inside the IP itself.	RO	0x0

usb0_m_I_main_QosGenerator_Priority

Priority register.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_usb0_m_I_main_QosGenerator	0xFFD16480	0xFFD16488

Offset: 0x8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MARK RO 0x1	Reserved														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						P1 RW 0x1		Reserved						P0 RW 0x0	

usb0_m_I_main_QosGenerator_Priority Fields

Bit	Name	Description	Access	Reset
31	MARK	Backward compatibility marker when 0.	RO	0x1
9:8	P1	In Programmable or Bandwidth Limiter mode, the priority level for read transactions. In Bandwidth regulator mode, the priority level when the used throughput is below the threshold. In Bandwidth Regulator mode, P1 should have a value equal or greater than P0.	RW	0x1

Bit	Name	Description	Access	Reset
1:0	P0	In Programmable or Bandwidth Limiter mode, the priority level for write transactions. In Bandwidth Regulator mode, the priority level when the used throughput is above the threshold. In Bandwidth Regulator mode, P0 should have a value equal or lower than P1.	RW	0x0

usb0_m_I_main_QosGenerator_Mode

Module Instance	Base Address	Register Address
i_noc_mpu_m0_usb0_m_I_main_QosGenerator	0xFFD16480	0xFFD1648C

Offset: 0xC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														MODE RW 0x3	

usb0_m_I_main_QosGenerator_Mode Fields

Bit	Name	Description	Access	Reset
1:0	MODE	0 = Programmable mode: a programmed priority is assigned to each read or write, 1 = Bandwidth Limiter Mode: a hard limit restricts throughput, 2 = Bypass mode: (<See SoC-specific QoS generator documentation>), 3 = Bandwidth Regulator mode: priority decreases when throughput exceeds a threshold.	RW	0x3

usb0_m_I_main_QosGenerator_Bandwidth

Module Instance	Base Address	Register Address
i_noc_mpu_m0_usb0_m_I_main_QosGenerator	0xFFD16480	0xFFD16490

Offset: 0x10

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					BANDWIDTH RW 0x100										

usb0_m_I_main_QosGenerator_Bandwidth Fields

Bit	Name	Description	Access	Reset
10:0	BANDWIDTH	In Bandwidth Limiter or Bandwidth Regulator mode, the bandwidth threshold is in units of 1/256th bytes per cycle. The formula to calculate this value is: Bandwidth = Throughput (Mbsp) * 256 / Clock Frequency. For example, 80 MBps on a 250 MHz interface has the value 0x0052 (BW = 80Mbps * 256/ 250MHz = 81.92, which is rounded to 0x0052).	RW	0x100

usb0_m_I_main_QosGenerator_Saturation

Module Instance	Base Address	Register Address
i_noc_mpu_m0_usb0_m_I_main_QosGenerator	0xFFD16480	0xFFD16494

Offset: 0x14

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						SATURATION RW 0x4									

usb0_m_I_main_QosGenerator_Saturation Fields

Bit	Name	Description	Access	Reset
9:0	SATURATION	In Bandwidth Limiter or Bandwidth Regulator mode, the maximum data count value, in units of 16 bytes. This determines the window of time over which bandwidth is measured. For example, to measure bandwidth within a 1000 cycle window on a 64-bit interface is value 0x1F4.	RW	0x4

usb0_m_I_main_QosGenerator_ExtControl

External inputs control.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_usb0_m_I_main_QosGenerator	0xFFD16480	0xFFD16498

Offset: 0x18

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													INTCLKEN RW 0x0	EXTTHREN RW 0x0	SOCKETQOSEN RW 0x0

usb0_m_I_main_QosGenerator_ExtControl Fields

Bit	Name	Description	Access	Reset
2	INTCLKEN	n/a	RW	0x0
1	EXTTHREN	n/a	RW	0x0

Bit	Name	Description	Access	Reset
0	SOCKETQOSEN	n/a	RW	0x0

noc_mpu_usb1_m_I_main_QosGenerator Address Map

Module Instance	Base Address	End Address
i_noc_mpu_m0_usb1_m_I_main_QosGenerator	0xFFD16500	0xFFD1657F

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
usb1_m_I_main_QosGenerator_Id_CoreId on page 7-755	0x0	32	RO	0x5DB54E04	
usb1_m_I_main_QosGenerator_Id_RevisionId on page 7-756	0x4	32	RO	0x129FF00	
usb1_m_I_main_QosGenerator_Priority on page 7-757	0x8	32	RW	0x80000100	Priority register.
usb1_m_I_main_QosGenerator_Mode on page 7-758	0xC	32	RW	0x3	
usb1_m_I_main_QosGenerator_Bandwidth on page 7-759	0x10	32	RW	0x100	
usb1_m_I_main_QosGenerator_Saturation on page 7-760	0x14	32	RW	0x4	
usb1_m_I_main_QosGenerator_ExtControl on page 7-761	0x18	32	RW	0x0	External inputs control.

noc_mpu_usb1_m_I_main_QosGenerator Summary

Base Address: 0xFFD16500

Register Address Offset	Bit Fields															
i_noc_mpu_m0_usb1_m_I_main_QosGener-ator																
usb1_m_I_main_QosGenerator_Id_CoreId 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	CORECHECKSUM RO 0x5DB54E															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CORECHECKSUM RO 0x5DB54E								CORETYPEID RO 0x4							
usb1_m_I_main_QosGenerator_Id_RevisionId 0x4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	FLEXNOCID RO 0x129FF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FLEXNOCID RO 0x129FF								USERID RO 0x0							
usb1_m_I_main_QosGenerator_Priority 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	MARK RO 0x1	Reserved														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						P1 RW 0x1		Reserved						P0 RW 0x0	
usb1_m_I_main_QosGenerator_Mode 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														MODE RW 0x3	
usb1_m_I_main_QosGenerator_Bandwidth 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved					BANDWIDTH RW 0x100										

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
usb1_m_I_main_QosGenerator_Saturation 0x14	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						SATURATION RW 0x4									
usb1_m_I_main_QosGenerator_ExtControl 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													INTCLKEN RW 0x0	EXTTHREN RW 0x0	SOCKETQOSEN RW 0x0

usb1_m_I_main_QosGenerator_Id_CoreId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_usb1_m_I_main_QosGenerator	0xFFD16500	0xFFD16500

Offset: 0x0

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CORECHECKSUM RO 0x5DB54E															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CORECHECKSUM RO 0x5DB54E								CORETYPEID RO 0x4							

usb1_m_I_main_QosGenerator_Id_CoreId Fields

Bit	Name	Description	Access	Reset
31:8	CORECHECKSUM	Field containing a checksum of the parameters of the IP.	RO	0x5DB54E
7:0	CORETYPEID	Field identifying the type of IP.	RO	0x4

usb1_m_I_main_QosGenerator_Id_RevisionId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_usb1_m_I_main_QosGenerator	0xFFD16500	0xFFD16504

Offset: 0x4

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLEXNOCID RO 0x129FF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLEXNOCID RO 0x129FF								USERID RO 0x0							

usb1_m_I_main_QosGenerator_Id_RevisionId Fields

Bit	Name	Description	Access	Reset
31:8	FLEXNOCID	Field containing the build revision of the software used to generate the IP HDL code.	RO	0x129FF
7:0	USERID	Field containing a user defined value, not used anywhere inside the IP itself.	RO	0x0

usb1_m_I_main_QosGenerator_Priority

Priority register.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_usb1_m_I_main_QosGenerator	0xFFD16500	0xFFD16508

Offset: 0x8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MARK RO 0x1	Reserved														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						P1 RW 0x1		Reserved						P0 RW 0x0	

usb1_m_I_main_QosGenerator_Priority Fields

Bit	Name	Description	Access	Reset
31	MARK	Backward compatibility marker when 0.	RO	0x1
9:8	P1	In Programmable or Bandwidth Limiter mode, the priority level for read transactions. In Bandwidth regulator mode, the priority level when the used throughput is below the threshold. In Bandwidth Regulator mode, P1 should have a value equal or greater than P0.	RW	0x1

Bit	Name	Description	Access	Reset
1:0	P0	In Programmable or Bandwidth Limiter mode, the priority level for write transactions. In Bandwidth Regulator mode, the priority level when the used throughput is above the threshold. In Bandwidth Regulator mode, P0 should have a value equal or lower than P1.	RW	0x0

usb1_m_I_main_QosGenerator_Mode

Module Instance	Base Address	Register Address
i_noc_mpu_m0_usb1_m_I_main_QosGenerator	0xFFD16500	0xFFD1650C

Offset: 0xC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														MODE RW 0x3	

usb1_m_I_main_QosGenerator_Mode Fields

Bit	Name	Description	Access	Reset
1:0	MODE	0 = Programmable mode: a programmed priority is assigned to each read or write, 1 = Bandwidth Limiter Mode: a hard limit restricts throughput, 2 = Bypass mode: (<See SoC-specific QoS generator documentation>), 3 = Bandwidth Regulator mode: priority decreases when throughput exceeds a threshold.	RW	0x3

usb1_m_I_main_QosGenerator_Bandwidth

Module Instance	Base Address	Register Address
i_noc_mpu_m0_usb1_m_I_main_QosGenerator	0xFFD16500	0xFFD16510

Offset: 0x10

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					BANDWIDTH RW 0x100										

usb1_m_I_main_QosGenerator_Bandwidth Fields

Bit	Name	Description	Access	Reset
10:0	BANDWIDTH	In Bandwidth Limiter or Bandwidth Regulator mode, the bandwidth threshold is in units of 1/256th bytes per cycle. The formula to calculate this value is: Bandwidth = Throughput (Mbsp) * 256 / Clock Frequency. For example, 80 MBps on a 250 MHz interface has the value 0x0052 (BW = 80Mbps * 256/ 250MHz = 81.92, which is rounded to 0x0052).	RW	0x100

usb1_m_I_main_QosGenerator_Saturation

Module Instance	Base Address	Register Address
i_noc_mpu_m0_usb1_m_I_main_QosGenerator	0xFFD16500	0xFFD16514

Offset: 0x14

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						SATURATION RW 0x4									

usb1_m_I_main_QosGenerator_Saturation Fields

Bit	Name	Description	Access	Reset
9:0	SATURATION	In Bandwidth Limiter or Bandwidth Regulator mode, the maximum data count value, in units of 16 bytes. This determines the window of time over which bandwidth is measured. For example, to measure bandwidth within a 1000 cycle window on a 64-bit interface is value 0x1F4.	RW	0x4

usb1_m_I_main_QosGenerator_ExtControl

External inputs control.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_usb1_m_I_main_QosGenerator	0xFFD16500	0xFFD16518

Offset: 0x18

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													INTCLKEN RW 0x0	EXTTHREN RW 0x0	SOCKETQOSEN RW 0x0

usb1_m_I_main_QosGenerator_ExtControl Fields

Bit	Name	Description	Access	Reset
2	INTCLKEN	n/a	RW	0x0
1	EXTTHREN	n/a	RW	0x0

Bit	Name	Description	Access	Reset
0	SOCKETQOSEN	n/a	RW	0x0

noc_mpu_nand_m_I_main_QosGenerator Address Map

Module Instance	Base Address	End Address
i_noc_mpu_m0_nand_m_I_main_QosGenerator	0xFFD16580	0xFFD165FF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
nand_m_I_main_QosGenerator_Id_CoreId on page 7-764	0x0	32	RO	0x260DE404	
nand_m_I_main_QosGenerator_Id_RevisionId on page 7-765	0x4	32	RO	0x129FF00	
nand_m_I_main_QosGenerator_Priority on page 7-766	0x8	32	RW	0x80000100	Priority register.
nand_m_I_main_QosGenerator_Mode on page 7-767	0xC	32	RW	0x3	
nand_m_I_main_QosGenerator_Bandwidth on page 7-768	0x10	32	RW	0x100	
nand_m_I_main_QosGenerator_Saturation on page 7-769	0x14	32	RW	0x4	
nand_m_I_main_QosGenerator_ExtControl on page 7-770	0x18	32	RW	0x0	External inputs control.

noc_mpu_nand_m_I_main_QosGenerator Summary

Base Address: 0xFFD16580

Register Address Offset	Bit Fields															
i_noc_mpu_m0_nand_m_I_main_QosGener-ator																
nand_m_I_main_QosGenerator_Id_CoreId 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	CORECHECKSUM RO 0x260DE4															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CORECHECKSUM RO 0x260DE4								CORETYPEID RO 0x4							
nand_m_I_main_QosGenerator_Id_RevisionId 0x4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	FLEXNOCID RO 0x129FF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FLEXNOCID RO 0x129FF								USERID RO 0x0							
nand_m_I_main_QosGenerator_Priority 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	MARK RO 0x1	Reserved														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						P1 RW 0x1		Reserved						P0 RW 0x0	
nand_m_I_main_QosGenerator_Mode 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														MODE RW 0x3	
nand_m_I_main_QosGenerator_Bandwidth 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved					BANDWIDTH RW 0x100										

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
nand_m_I_main_QosGenerator_Saturation 0x14	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						SATURATION RW 0x4									
nand_m_I_main_QosGenerator_ExtControl 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													INTCLKEN RW 0x0	EXTTHREN RW 0x0	SOCKETQOSEN RW 0x0

nand_m_I_main_QosGenerator_Id_CoreId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_nand_m_I_main_QosGenerator	0xFFD16580	0xFFD16580

Offset: 0x0

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CORECHECKSUM RO 0x260DE4															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CORECHECKSUM RO 0x260DE4								CORETYPEID RO 0x4							

nand_m_I_main_QosGenerator_Id_CoreId Fields

Bit	Name	Description	Access	Reset
31:8	CORECHECKSUM	Field containing a checksum of the parameters of the IP.	RO	0x260DE4
7:0	CORETYPEID	Field identifying the type of IP.	RO	0x4

nand_m_I_main_QosGenerator_Id_RevisionId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_nand_m_I_main_QosGenerator	0xFFD16580	0xFFD16584

Offset: 0x4

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLEXNOCID RO 0x129FF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLEXNOCID RO 0x129FF								USERID RO 0x0							

nand_m_I_main_QosGenerator_Id_RevisionId Fields

Bit	Name	Description	Access	Reset
31:8	FLEXNOCID	Field containing the build revision of the software used to generate the IP HDL code.	RO	0x129FF
7:0	USERID	Field containing a user defined value, not used anywhere inside the IP itself.	RO	0x0

nand_m_I_main_QosGenerator_Priority

Priority register.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_nand_m_I_main_QosGenerator	0xFFD16580	0xFFD16588

Offset: 0x8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MARK RO 0x1	Reserved														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						P1 RW 0x1		Reserved						P0 RW 0x0	

nand_m_I_main_QosGenerator_Priority Fields

Bit	Name	Description	Access	Reset
31	MARK	Backward compatibility marker when 0.	RO	0x1
9:8	P1	In Programmable or Bandwidth Limiter mode, the priority level for read transactions. In Bandwidth regulator mode, the priority level when the used throughput is below the threshold. In Bandwidth Regulator mode, P1 should have a value equal or greater than P0.	RW	0x1

Bit	Name	Description	Access	Reset
1:0	P0	In Programmable or Bandwidth Limiter mode, the priority level for write transactions. In Bandwidth Regulator mode, the priority level when the used throughput is above the threshold. In Bandwidth Regulator mode, P0 should have a value equal or lower than P1.	RW	0x0

nand_m_I_main_QosGenerator_Mode

Module Instance	Base Address	Register Address
i_noc_mpu_m0_nand_m_I_main_QosGenerator	0xFFD16580	0xFFD1658C

Offset: 0xC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														MODE RW 0x3	

nand_m_I_main_QosGenerator_Mode Fields

Bit	Name	Description	Access	Reset
1:0	MODE	0 = Programmable mode: a programmed priority is assigned to each read or write, 1 = Bandwidth Limiter Mode: a hard limit restricts throughput, 2 = Bypass mode: (<See SoC-specific QoS generator documentation>), 3 = Bandwidth Regulator mode: priority decreases when throughput exceeds a threshold.	RW	0x3

nand_m_I_main_QosGenerator_Bandwidth

Module Instance	Base Address	Register Address
i_noc_mpu_m0_nand_m_I_main_QosGenerator	0xFFD16580	0xFFD16590

Offset: 0x10

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					BANDWIDTH RW 0x100										

nand_m_I_main_QosGenerator_Bandwidth Fields

Bit	Name	Description	Access	Reset
10:0	BANDWIDTH	In Bandwidth Limiter or Bandwidth Regulator mode, the bandwidth threshold in units of 1/256th bytes per cycle. For example, 80 MBps on a 250 MHz interface is value 0x0052.	RW	0x100

nand_m_I_main_QosGenerator_Saturation

Module Instance	Base Address	Register Address
i_noc_mpu_m0_nand_m_I_main_QosGenerator	0xFFD16580	0xFFD16594

Offset: 0x14

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						SATURATION RW 0x4									

nand_m_I_main_QosGenerator_Saturation Fields

Bit	Name	Description	Access	Reset
9:0	SATURATION	In Bandwidth Limiter or Bandwidth Regulator mode, the maximum data count value, in units of 16 bytes. This determines the window of time over which bandwidth is measured. For example, to measure bandwidth within a 1000 cycle window on a 64-bit interface is value 0x1F4.	RW	0x4

nand_m_I_main_QosGenerator_ExtControl

External inputs control.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_nand_m_I_main_QosGenerator	0xFFD16580	0xFFD16598

Offset: 0x18

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													INTCLKEN RW 0x0	EXTTHREN RW 0x0	SOCKETQOSEN RW 0x0

nand_m_I_main_QosGenerator_ExtControl Fields

Bit	Name	Description	Access	Reset
2	INTCLKEN	n/a	RW	0x0
1	EXTTHREN	n/a	RW	0x0

Bit	Name	Description	Access	Reset
0	SOCKETQOSEN	n/a	RW	0x0

noc_mpu_sdmmc_m_I_main_QosGenerator Address Map

Module Instance	Base Address	End Address
i_noc_mpu_m0_sdmmc_m_I_main_QosGenerator	0xFFD16600	0xFFD1667F

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
sdmmc_m_I_main_QosGenerator_Id_CoreId on page 7-773	0x0	32	RO	0x72338104	
sdmmc_m_I_main_QosGenerator_Id_RevisionId on page 7-774	0x4	32	RO	0x129FF00	
sdmmc_m_I_main_QosGenerator_Priority on page 7-775	0x8	32	RW	0x80000100	Priority register.
sdmmc_m_I_main_QosGenerator_Mode on page 7-776	0xC	32	RW	0x3	
sdmmc_m_I_main_QosGenerator_Bandwidth on page 7-777	0x10	32	RW	0x100	
sdmmc_m_I_main_QosGenerator_Saturation on page 7-778	0x14	32	RW	0x4	
sdmmc_m_I_main_QosGenerator_ExtControl on page 7-779	0x18	32	RW	0x0	External inputs control.

noc_mpu_sdmmc_m_I_main_QosGenerator Summary

Base Address: 0xFFD16600

Register Address Offset	Bit Fields															
i_noc_mpu_m0_sdmmc_m_I_main_QosGenerator																
sdmmc_m_I_main_QosGenerator_Id_CoreId 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	CORECHECKSUM RO 0x723381															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CORECHECKSUM RO 0x723381								CORETYPEID RO 0x4							
sdmmc_m_I_main_QosGenerator_Id_RevisionId 0x4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	FLEXNOCID RO 0x129FF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FLEXNOCID RO 0x129FF								USERID RO 0x0							
sdmmc_m_I_main_QosGenerator_Priority 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	MARK RO 0x1	Reserved														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						P1 RW 0x1		Reserved						P0 RW 0x0	
sdmmc_m_I_main_QosGenerator_Mode 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														MODE RW 0x3	
sdmmc_m_I_main_QosGenerator_Bandwidth 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved					BANDWIDTH RW 0x100										

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
sdmmc_m_I_main_QosGenerator_Saturation 0x14	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						SATURATION RW 0x4									
sdmmc_m_I_main_QosGenerator_ExtControl 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													INTCLKEN RW 0x0	EXTTHREN RW 0x0	SOCKETQOSEN RW 0x0

sdmmc_m_I_main_QosGenerator_Id_CoreId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_sdmmc_m_I_main_QosGenerator	0xFFD16600	0xFFD16600

Offset: 0x0

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CORECHECKSUM RO 0x723381															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CORECHECKSUM RO 0x723381								CORETYPEID RO 0x4							

sdmmc_m_l_main_QosGenerator_Id_CoreId Fields

Bit	Name	Description	Access	Reset
31:8	CORECHECKSUM	Field containing a checksum of the parameters of the IP.	RO	0x723381
7:0	CORETYPEID	Field identifying the type of IP.	RO	0x4

sdmmc_m_l_main_QosGenerator_Id_RevisionId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_sdmmc_m_I_main_QosGenerator	0xFFD16600	0xFFD16604

Offset: 0x4

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLEXNOCID RO 0x129FF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLEXNOCID RO 0x129FF								USERID RO 0x0							

sdmmc_m_l_main_QosGenerator_Id_RevisionId Fields

Bit	Name	Description	Access	Reset
31:8	FLEXNOCID	Field containing the build revision of the software used to generate the IP HDL code.	RO	0x129FF
7:0	USERID	Field containing a user defined value, not used anywhere inside the IP itself.	RO	0x0

sdmmc_m_I_main_QosGenerator_Priority

Priority register.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_sdmmc_m_I_main_QosGenerator	0xFFD16600	0xFFD16608

Offset: 0x8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MARK RO 0x1	Reserved														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						P1 RW 0x1		Reserved						P0 RW 0x0	

sdmmc_m_I_main_QosGenerator_Priority Fields

Bit	Name	Description	Access	Reset
31	MARK	Backward compatibility marker when 0.	RO	0x1
9:8	P1	In Programmable or Bandwidth Limiter mode, the priority level for read transactions. In Bandwidth regulator mode, the priority level when the used throughput is below the threshold. In Bandwidth Regulator mode, P1 should have a value equal or greater than P0.	RW	0x1

Bit	Name	Description	Access	Reset
1:0	P0	In Programmable or Bandwidth Limiter mode, the priority level for write transactions. In Bandwidth Regulator mode, the priority level when the used throughput is above the threshold. In Bandwidth Regulator mode, P0 should have a value equal or lower than P1.	RW	0x0

sdmmc_m_I_main_QosGenerator_Mode

Module Instance	Base Address	Register Address
i_noc_mpu_m0_sdmmc_m_I_main_QosGenerator	0xFFD16600	0xFFD1660C

Offset: 0xC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														MODE RW 0x3	

sdmmc_m_I_main_QosGenerator_Mode Fields

Bit	Name	Description	Access	Reset
1:0	MODE	0 = Programmable mode: a programmed priority is assigned to each read or write, 1 = Bandwidth Limiter Mode: a hard limit restricts throughput, 2 = Bypass mode: (<See SoC-specific QoS generator documentation>), 3 = Bandwidth Regulator mode: priority decreases when throughput exceeds a threshold.	RW	0x3

sdmmc_m_I_main_QosGenerator_Bandwidth

Module Instance	Base Address	Register Address
i_noc_mpu_m0_sdmmc_m_I_main_QosGenerator	0xFFD16600	0xFFD16610

Offset: 0x10

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					BANDWIDTH RW 0x100										

sdmmc_m_I_main_QosGenerator_Bandwidth Fields

Bit	Name	Description	Access	Reset
10:0	BANDWIDTH	In Bandwidth Limiter or Bandwidth Regulator mode, the bandwidth threshold is in units of 1/256th bytes per cycle. The formula to calculate this value is: Bandwidth = Throughput (Mbsp) * 256 / Clock Frequency. For example, 80 MBps on a 250 MHz interface has the value 0x0052 (BW = 80Mbps * 256/ 250MHz = 81.92, which is rounded to 0x0052).	RW	0x100

sdmmc_m_I_main_QosGenerator_Saturation

Module Instance	Base Address	Register Address
i_noc_mpu_m0_sdmmc_m_I_main_QosGenerator	0xFFD16600	0xFFD16614

Offset: 0x14

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						SATURATION RW 0x4									

sdmmc_m_I_main_QosGenerator_Saturation Fields

Bit	Name	Description	Access	Reset
9:0	SATURATION	In Bandwidth Limiter or Bandwidth Regulator mode, the maximum data count value, in units of 16 bytes. This determines the window of time over which bandwidth is measured. For example, to measure bandwidth within a 1000 cycle window on a 64-bit interface is value 0x1F4.	RW	0x4

sdmmc_m_I_main_QosGenerator_ExtControl

External inputs control.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_sdmmc_m_I_main_QosGenerator	0xFFD16600	0xFFD16618

Offset: 0x18

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													INTCLKEN RW 0x0	EXTTHREN RW 0x0	SOCKETQOSEN RW 0x0

sdmmc_m_I_main_QosGenerator_ExtControl Fields

Bit	Name	Description	Access	Reset
2	INTCLKEN	n/a	RW	0x0
1	EXTTHREN	n/a	RW	0x0

Bit	Name	Description	Access	Reset
0	SOCKETQOSEN	n/a	RW	0x0

noc_mpu_fpga2sdram0_axi32_I_main_QosGenerator Address Map

Module Instance	Base Address	End Address
i_noc_mpu_m0_fpga2sdram0_axi32_I_main_QosGenerator	0xFFD16680	0xFFD166FF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
fpga2sdram0_axi32_I_main_QosGenerator_Id_CoreId on page 7-782	0x0	32	RO	0x88AEF404	
fpga2sdram0_axi32_I_main_QosGenerator_Id_RevisionId on page 7-783	0x4	32	RO	0x129FF00	
fpga2sdram0_axi32_I_main_QosGenerator_Priority on page 7-784	0x8	32	RW	0x80000200	Priority register.
fpga2sdram0_axi32_I_main_QosGenerator_Mode on page 7-785	0xC	32	RW	0x3	
fpga2sdram0_axi32_I_main_QosGenerator_Bandwidth on page 7-786	0x10	32	RW	0x280	
fpga2sdram0_axi32_I_main_QosGenerator_Saturation on page 7-787	0x14	32	RW	0x8	

Register	Offset	Width	Access	Reset Value	Description
fpga2sdram0_axi32_I_main_QosGenerator_ExtControl on page 7-788	0x18	32	RW	0x0	External inputs control.

noc_mpu_fpga2sdram0_axi32_I_main_QosGenerator Summary

Base Address: 0xFFD16680

Register Address Offset	Bit Fields															
i_noc_mpu_m0_fpga2sdram0_axi32_I_main_QosGenerator																
fpga2sdram0_axi32_I_main_QosGenerator_Id_CoreId 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	CORECHECKSUM RO 0x88AEF4															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CORECHECKSUM RO 0x88AEF4								CORETYPEID RO 0x4							
fpga2sdram0_axi32_I_main_QosGenerator_Id_RevisionId 0x4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	FLEXNOCID RO 0x129FF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FLEXNOCID RO 0x129FF								USERID RO 0x0							
fpga2sdram0_axi32_I_main_QosGenerator_Priority 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	MARK RO 0x1	Reserved														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						P1 RW 0x2		Reserved						P0 RW 0x0	

Register Address Offset	Bit Fields															
fpga2sdram0_axi32_I_main_QosGenerator_Mode 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														MODE RW 0x3	
fpga2sdram0_axi32_I_main_QosGenerator_Bandwidth 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						BANDWIDTH RW 0x280									
fpga2sdram0_axi32_I_main_QosGenerator_Saturation 0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						SATURATION RW 0x8									
fpga2sdram0_axi32_I_main_QosGenerator_ExtControl 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													INTCLKEN RW 0x0	EXTTHREN RW 0x0	SOCKETQOSEN RW 0x0

fpga2sdram0_axi32_I_main_QosGenerator_Id_CoreId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2sdram0_axi32_I_main_QosGenerator	0xFFD16680	0xFFD16680

Offset: 0x0

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CORECHECKSUM RO 0x88AEF4															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CORECHECKSUM RO 0x88AEF4								CORETYPEID RO 0x4							

fpga2sdram0_axi32_I_main_QosGenerator_Id_CoreId Fields

Bit	Name	Description	Access	Reset
31:8	CORECHECKSUM	Field containing a checksum of the parameters of the IP.	RO	0x88AEF4
7:0	CORETYPEID	Field identifying the type of IP.	RO	0x4

fpga2sdram0_axi32_I_main_QosGenerator_Id_RevisionId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2sdram0_axi32_I_main_QosGenerator	0xFFD16680	0xFFD16684

Offset: 0x4

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLEXNOCID RO 0x129FF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLEXNOCID RO 0x129FF								USERID RO 0x0							

fpga2sdram0_axi32_I_main_QosGenerator_Id_RevisionId Fields

Bit	Name	Description	Access	Reset
31:8	FLEXNOCID	Field containing the build revision of the software used to generate the IP HDL code.	RO	0x129FF
7:0	USERID	Field containing a user defined value, not used anywhere inside the IP itself.	RO	0x0

fpga2sdram0_axi32_I_main_QosGenerator_Priority

Priority register.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2sdram0_axi32_I_main_QosGenerator	0xFFD16680	0xFFD16688

Offset: 0x8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MARK RO 0x1	Reserved														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						P1 RW 0x2		Reserved						P0 RW 0x0	

fpga2sdram0_axi32_I_main_QosGenerator_Priority Fields

Bit	Name	Description	Access	Reset
31	MARK	Backward compatibility marker when 0.	RO	0x1

Bit	Name	Description	Access	Reset
9:8	P1	In Programmable or Bandwidth Limiter mode, the priority level for read transactions. In Bandwidth regulator mode, the priority level when the used throughput is below the threshold. In Bandwidth Regulator mode, P1 should have a value equal or greater than P0.	RW	0x2
1:0	P0	In Programmable or Bandwidth Limiter mode, the priority level for write transactions. In Bandwidth Regulator mode, the priority level when the used throughput is above the threshold. In Bandwidth Regulator mode, P0 should have a value equal or lower than P1.	RW	0x0

fpga2sdram0_axi32_I_main_QosGenerator_Mode

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2sdram0_axi32_I_main_QosGenerator	0xFFD16680	0xFFD1668C

Offset: 0xC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														MODE RW 0x3	

fpga2sdram0_axi32_I_main_QosGenerator_Mode Fields

Bit	Name	Description	Access	Reset
1:0	MODE	0 = Programmable mode: a programmed priority is assigned to each read or write, 1 = Bandwidth Limiter Mode: a hard limit restricts throughput, 2 = Bypass mode: (<See SoC-specific QoS generator documentation>), 3 = Bandwidth Regulator mode: priority decreases when throughput exceeds a threshold.	RW	0x3

fpga2sdram0_axi32_I_main_QosGenerator_Bandwidth

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2sdram0_axi32_I_main_QosGenerator	0xFFD16680	0xFFD16690

Offset: 0x10

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					BANDWIDTH RW 0x280										

fpga2sdram0_axi32_I_main_QosGenerator_Bandwidth Fields

Bit	Name	Description	Access	Reset
10:0	BANDWIDTH	In Bandwidth Limiter or Bandwidth Regulator mode, the bandwidth threshold is in units of 1/256th bytes per cycle. The formula to calculate this value is: Bandwidth = Throughput (Mbsp) * 256 / Clock Frequency. For example, 80 MBps on a 250 MHz interface has the value 0x0052 (BW = 80Mbps * 256/ 250MHz = 81.92, which is rounded to 0x0052).	RW	0x280

fpga2sdram0_axi32_I_main_QosGenerator_Saturation

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2sdram0_axi32_I_main_QosGenerator	0xFFD16680	0xFFD16694

Offset: 0x14

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						SATURATION RW 0x8									

fpga2sdram0_axi32_I_main_QosGenerator_Saturation Fields

Bit	Name	Description	Access	Reset
9:0	SATURATION	In Bandwidth Limiter or Bandwidth Regulator mode, the maximum data count value, in units of 16 bytes. This determines the window of time over which bandwidth is measured. For example, to measure bandwidth within a 1000 cycle window on a 64-bit interface is value 0x1F4.	RW	0x8

fpga2sdram0_axi32_I_main_QosGenerator_ExtControl

External inputs control.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2sdram0_axi32_I_main_QosGenerator	0xFFD16680	0xFFD16698

Offset: 0x18

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													INTCLKEN RW 0x0	EXTTHREN RW 0x0	SOCKETQOSEN RW 0x0

fpga2sdram0_axi32_I_main_QosGenerator_ExtControl Fields

Bit	Name	Description	Access	Reset
2	INTCLKEN	n/a	RW	0x0
1	EXTTHREN	n/a	RW	0x0

Bit	Name	Description	Access	Reset
0	SOCKETQOSEN	n/a	RW	0x0

noc_mpu_fpga2sdram0_axi64_I_main_QosGenerator Address Map

Module Instance	Base Address	End Address
i_noc_mpu_m0_fpga2sdram0_axi64_I_main_QosGenerator	0xFFD16700	0xFFD1677F

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
fpga2sdram0_axi64_I_main_QosGenerator_Id_CoreId on page 7-791	0x0	32	RO	0x130F6604	
fpga2sdram0_axi64_I_main_QosGenerator_Id_RevisionId on page 7-792	0x4	32	RO	0x129FF00	
fpga2sdram0_axi64_I_main_QosGenerator_Priority on page 7-793	0x8	32	RW	0x80000200	Priority register.
fpga2sdram0_axi64_I_main_QosGenerator_Mode on page 7-794	0xC	32	RW	0x3	
fpga2sdram0_axi64_I_main_QosGenerator_Bandwidth on page 7-795	0x10	32	RW	0x780	
fpga2sdram0_axi64_I_main_QosGenerator_Saturation on page 7-796	0x14	32	RW	0x8	

Register	Offset	Width	Access	Reset Value	Description
fpga2sdram0_axi64_I_main_QosGenerator_ExtControl on page 7-797	0x18	32	RW	0x0	External inputs control.

noc_mpu_fpga2sdram0_axi64_I_main_QosGenerator Summary

Base Address: 0xFFD16700

Register Address Offset	Bit Fields															
i_noc_mpu_m0_fpga2sdram0_axi64_I_main_QosGenerator																
fpga2sdram0_axi64_I_main_QosGenerator_Id_CoreId 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	CORECHECKSUM RO 0x130F66															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CORECHECKSUM RO 0x130F66								CORETYPEID RO 0x4							
fpga2sdram0_axi64_I_main_QosGenerator_Id_RevisionId 0x4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	FLEXNOCID RO 0x129FF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FLEXNOCID RO 0x129FF								USERID RO 0x0							
fpga2sdram0_axi64_I_main_QosGenerator_Priority 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	MARK RO 0x1	Reserved														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						P1 RW 0x2		Reserved						P0 RW 0x0	

Register Address Offset	Bit Fields															
fpga2sdram0_axi64_I_main_QosGenerator_Mode 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														MODE RW 0x3	
fpga2sdram0_axi64_I_main_QosGenerator_Bandwidth 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				BANDWIDTH RW 0x780											
fpga2sdram0_axi64_I_main_QosGenerator_Saturation 0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						SATURATION RW 0x8									
fpga2sdram0_axi64_I_main_QosGenerator_ExtControl 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													INTCLKEN RW 0x0	EXTTHREN RW 0x0	SOCKETQOSEN RW 0x0

fpga2sdram0_axi64_I_main_QosGenerator_Id_CoreId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2sdram0_axi64_I_main_QosGenerator	0xFFD16700	0xFFD16700

Offset: 0x0

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CORECHECKSUM RO 0x130F66															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CORECHECKSUM RO 0x130F66								CORETYPEID RO 0x4							

fpga2sdram0_axi64_I_main_QosGenerator_Id_CoreId Fields

Bit	Name	Description	Access	Reset
31:8	CORECHECKSUM	Field containing a checksum of the parameters of the IP.	RO	0x130F66
7:0	CORETYPEID	Field identifying the type of IP.	RO	0x4

fpga2sdram0_axi64_I_main_QosGenerator_Id_RevisionId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2sdram0_axi64_I_main_QosGenerator	0xFFD16700	0xFFD16704

Offset: 0x4

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLEXNOCID RO 0x129FF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLEXNOCID RO 0x129FF								USERID RO 0x0							

fpga2sdram0_axi64_I_main_QosGenerator_Id_RevisionId Fields

Bit	Name	Description	Access	Reset
31:8	FLEXNOCID	Field containing the build revision of the software used to generate the IP HDL code.	RO	0x129FF
7:0	USERID	Field containing a user defined value, not used anywhere inside the IP itself.	RO	0x0

fpga2sdram0_axi64_I_main_QosGenerator_Priority

Priority register.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2sdram0_axi64_I_main_QosGenerator	0xFFD16700	0xFFD16708

Offset: 0x8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MARK RO 0x1	Reserved														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						P1 RW 0x2		Reserved						P0 RW 0x0	

fpga2sdram0_axi64_I_main_QosGenerator_Priority Fields

Bit	Name	Description	Access	Reset
31	MARK	Backward compatibility marker when 0.	RO	0x1

Bit	Name	Description	Access	Reset
9:8	P1	In Programmable or Bandwidth Limiter mode, the priority level for read transactions. In Bandwidth regulator mode, the priority level when the used throughput is below the threshold. In Bandwidth Regulator mode, P1 should have a value equal or greater than P0.	RW	0x2
1:0	P0	In Programmable or Bandwidth Limiter mode, the priority level for write transactions. In Bandwidth Regulator mode, the priority level when the used throughput is above the threshold. In Bandwidth Regulator mode, P0 should have a value equal or lower than P1.	RW	0x0

fpga2sdram0_axi64_I_main_QosGenerator_Mode

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2sdram0_axi64_I_main_QosGenerator	0xFFD16700	0xFFD1670C

Offset: 0xC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														MODE RW 0x3	

fpga2sdram0_axi64_I_main_QosGenerator_Mode Fields

Bit	Name	Description	Access	Reset
1:0	MODE	0 = Programmable mode: a programmed priority is assigned to each read or write, 1 = Bandwidth Limiter Mode: a hard limit restricts throughput, 2 = Bypass mode: (<See SoC-specific QoS generator documentation>), 3 = Bandwidth Regulator mode: priority decreases when throughput exceeds a threshold.	RW	0x3

fpga2sdram0_axi64_I_main_QosGenerator_Bandwidth

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2sdram0_axi64_I_main_QosGenerator	0xFFD16700	0xFFD16710

Offset: 0x10

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				BANDWIDTH RW 0x780											

fpga2sdram0_axi64_I_main_QosGenerator_Bandwidth Fields

Bit	Name	Description	Access	Reset
11:0	BANDWIDTH	In Bandwidth Limiter or Bandwidth Regulator mode, the bandwidth threshold is in units of 1/256th bytes per cycle. The formula to calculate this value is: Bandwidth = Throughput (Mbsp) * 256 / Clock Frequency. For example, 80 MBps on a 250 MHz interface has the value 0x0052 (BW = 80Mbps * 256/ 250MHz = 81.92, which is rounded to 0x0052).	RW	0x780

fpga2sdram0_axi64_I_main_QosGenerator_Saturation

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2sdram0_axi64_I_main_QosGenerator	0xFFD16700	0xFFD16714

Offset: 0x14

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						SATURATION RW 0x8									

fpga2sdram0_axi64_I_main_QosGenerator_Saturation Fields

Bit	Name	Description	Access	Reset
9:0	SATURATION	In Bandwidth Limiter or Bandwidth Regulator mode, the maximum data count value, in units of 16 bytes. This determines the window of time over which bandwidth is measured. For example, to measure bandwidth within a 1000 cycle window on a 64-bit interface is value 0x1F4.	RW	0x8

fpga2sdram0_axi64_I_main_QosGenerator_ExtControl

External inputs control.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2sdram0_axi64_I_main_QosGenerator	0xFFD16700	0xFFD16718

Offset: 0x18

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													INTCLKEN RW 0x0	EXTTHREN RW 0x0	SOCKETQOSEN RW 0x0

fpga2sdram0_axi64_I_main_QosGenerator_ExtControl Fields

Bit	Name	Description	Access	Reset
2	INTCLKEN	n/a	RW	0x0
1	EXTTHREN	n/a	RW	0x0

Bit	Name	Description	Access	Reset
0	SOCKETQOSEN	n/a	RW	0x0

noc_mpu_fpga2sdram0_axi128_I_main_QosGenerator Address Map

Module Instance	Base Address	End Address
i_noc_mpu_m0_fpga2sdram0_axi128_I_main_QosGenerator	0xFFD16780	0xFFD167FF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
fpga2sdram0_axi128_I_main_QosGenerator_Id_CoreId on page 7-800	0x0	32	RO	0x721F804	
fpga2sdram0_axi128_I_main_QosGenerator_Id_RevisionId on page 7-801	0x4	32	RO	0x129FF00	
fpga2sdram0_axi128_I_main_QosGenerator_Priority on page 7-802	0x8	32	RW	0x80000200	Priority register.
fpga2sdram0_axi128_I_main_QosGenerator_Mode on page 7-803	0xC	32	RW	0x3	
fpga2sdram0_axi128_I_main_QosGenerator_Bandwidth on page 7-804	0x10	32	RW	0xC80	
fpga2sdram0_axi128_I_main_QosGenerator_Saturation on page 7-805	0x14	32	RW	0x8	

Register	Offset	Width	Access	Reset Value	Description
fpga2sdram0_axi128_I_main_QosGenerator_ExtControl on page 7-806	0x18	32	RW	0x0	External inputs control.

noc_mpu_fpga2sdram0_axi128_I_main_QosGenerator Summary

Base Address: 0xFFD16780

Register Address Offset	Bit Fields															
i_noc_mpu_m0_fpga2sdram0_axi128_I_main_QosGenerator																
fpga2sdram0_axi128_I_main_QosGenerator_Id_CoreId 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	CORECHECKSUM RO 0x721F8															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CORECHECKSUM RO 0x721F8								CORETYPEID RO 0x4							
fpga2sdram0_axi128_I_main_QosGenerator_Id_RevisionId 0x4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	FLEXNOCID RO 0x129FF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FLEXNOCID RO 0x129FF								USERID RO 0x0							
fpga2sdram0_axi128_I_main_QosGenerator_Priority 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	MARK RO 0x1	Reserved														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						P1 RW 0x2		Reserved						P0 RW 0x0	

Register Address Offset	Bit Fields															
fpga2sdram0_axi128_I_main_QosGenerator_Mode 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														MODE RW 0x3	
fpga2sdram0_axi128_I_main_QosGenerator_Bandwidth 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved			BANDWIDTH RW 0xC80												
fpga2sdram0_axi128_I_main_QosGenerator_Saturation 0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						SATURATION RW 0x8									
fpga2sdram0_axi128_I_main_QosGenerator_ExtControl 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													INTCLKEN RW 0x0	EXTTHREN RW 0x0	SOCKETQOSEN RW 0x0

fpga2sdram0_axi128_I_main_QosGenerator_Id_CoreId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2sdram0_axi128_I_main_QosGenerator	0xFFD16780	0xFFD16780

Offset: 0x0

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CORECHECKSUM RO 0x721F8															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CORECHECKSUM RO 0x721F8								CORETYPEID RO 0x4							

fpga2sdram0_axi128_I_main_QosGenerator_Id_CoreId Fields

Bit	Name	Description	Access	Reset
31:8	CORECHECKSUM	Field containing a checksum of the parameters of the IP.	RO	0x721F8
7:0	CORETYPEID	Field identifying the type of IP.	RO	0x4

fpga2sdram0_axi128_I_main_QosGenerator_Id_RevisionId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2sdram0_axi128_I_main_QosGenerator	0xFFD16780	0xFFD16784

Offset: 0x4

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLEXNOCID RO 0x129FF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLEXNOCID RO 0x129FF								USERID RO 0x0							

fpga2sdram0_axi128_I_main_QosGenerator_Id_RevisionId Fields

Bit	Name	Description	Access	Reset
31:8	FLEXNOCID	Field containing the build revision of the software used to generate the IP HDL code.	RO	0x129FF
7:0	USERID	Field containing a user defined value, not used anywhere inside the IP itself.	RO	0x0

fpga2sdram0_axi128_I_main_QosGenerator_Priority

Priority register.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2sdram0_axi128_I_main_QosGenerator	0xFFD16780	0xFFD16788

Offset: 0x8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MARK RO 0x1	Reserved														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						P1 RW 0x2		Reserved						P0 RW 0x0	

fpga2sdram0_axi128_I_main_QosGenerator_Priority Fields

Bit	Name	Description	Access	Reset
31	MARK	Backward compatibility marker when 0.	RO	0x1

Bit	Name	Description	Access	Reset
9:8	P1	In Programmable or Bandwidth Limiter mode, the priority level for read transactions. In Bandwidth regulator mode, the priority level when the used throughput is below the threshold. In Bandwidth Regulator mode, P1 should have a value equal or greater than P0.	RW	0x2
1:0	P0	In Programmable or Bandwidth Limiter mode, the priority level for write transactions. In Bandwidth Regulator mode, the priority level when the used throughput is above the threshold. In Bandwidth Regulator mode, P0 should have a value equal or lower than P1.	RW	0x0

fpga2sdram0_axi128_I_main_QosGenerator_Mode

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2sdram0_axi128_I_main_QosGenerator	0xFFD16780	0xFFD1678C

Offset: 0xC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														MODE RW 0x3	

fpga2sdram0_axi128_I_main_QosGenerator_Mode Fields

Bit	Name	Description	Access	Reset
1:0	MODE	0 = Programmable mode: a programmed priority is assigned to each read or write, 1 = Bandwidth Limiter Mode: a hard limit restricts throughput, 2 = Bypass mode: (<See SoC-specific QoS generator documentation>), 3 = Bandwidth Regulator mode: priority decreases when throughput exceeds a threshold.	RW	0x3

fpga2sdram0_axi128_I_main_QosGenerator_Bandwidth

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2sdram0_axi128_I_main_QosGenerator	0xFFD16780	0xFFD16790

Offset: 0x10

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			BANDWIDTH RW 0xC80												

fpga2sdram0_axi128_I_main_QosGenerator_Bandwidth Fields

Bit	Name	Description	Access	Reset
12:0	BANDWIDTH	In Bandwidth Limiter or Bandwidth Regulator mode, the bandwidth threshold is in units of 1/256th bytes per cycle. The formula to calculate this value is: Bandwidth = Throughput (Mbsp) * 256 / Clock Frequency. For example, 80 MBps on a 250 MHz interface has the value 0x0052 (BW = 80Mbps * 256/ 250MHz = 81.92, which is rounded to 0x0052).	RW	0xC80

fpga2sdram0_axi128_I_main_QosGenerator_Saturation

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2sdram0_axi128_I_main_QosGenerator	0xFFD16780	0xFFD16794

Offset: 0x14

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						SATURATION RW 0x8									

fpga2sdram0_axi128_I_main_QosGenerator_Saturation Fields

Bit	Name	Description	Access	Reset
9:0	SATURATION	In Bandwidth Limiter or Bandwidth Regulator mode, the maximum data count value, in units of 16 bytes. This determines the window of time over which bandwidth is measured. For example, to measure bandwidth within a 1000 cycle window on a 64-bit interface is value 0x1F4.	RW	0x8

fpga2sdram0_axi128_I_main_QosGenerator_ExtControl

External inputs control.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2sdram0_axi128_I_main_QosGenerator	0xFFD16780	0xFFD16798

Offset: 0x18

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													INTCLKEN RW 0x0	EXTTHREN RW 0x0	SOCKETQOSEN RW 0x0

fpga2sdram0_axi128_I_main_QosGenerator_ExtControl Fields

Bit	Name	Description	Access	Reset
2	INTCLKEN	n/a	RW	0x0
1	EXTTHREN	n/a	RW	0x0

Bit	Name	Description	Access	Reset
0	SOCKETQOSEN	n/a	RW	0x0

noc_mpu_fpga2sdram1_axi32_I_main_QosGenerator Address Map

Module Instance	Base Address	End Address
i_noc_mpu_m0_fpga2sdram1_axi32_I_main_QosGenerator	0xFFD16800	0xFFD1687F

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
fpga2sdram1_axi32_I_main_QosGenerator_Id_CoreId on page 7-809	0x0	32	RO	0xB500B904	
fpga2sdram1_axi32_I_main_QosGenerator_Id_RevisionId on page 7-810	0x4	32	RO	0x129FF00	
fpga2sdram1_axi32_I_main_QosGenerator_Priority on page 7-811	0x8	32	RW	0x80000200	Priority register.
fpga2sdram1_axi32_I_main_QosGenerator_Mode on page 7-812	0xC	32	RW	0x3	
fpga2sdram1_axi32_I_main_QosGenerator_Bandwidth on page 7-813	0x10	32	RW	0x280	
fpga2sdram1_axi32_I_main_QosGenerator_Saturation on page 7-814	0x14	32	RW	0x8	

Register	Offset	Width	Access	Reset Value	Description
fpga2sdram1_axi32_I_main_QosGenerator_ExtControl on page 7-815	0x18	32	RW	0x0	External inputs control.

noc_mpu_fpga2sdram1_axi32_I_main_QosGenerator Summary

Base Address: 0xFFD16800

Register Address Offset	Bit Fields															
i_noc_mpu_m0_fpga2sdram1_axi32_I_main_QosGenerator																
fpga2sdram1_axi32_I_main_QosGenerator_Id_CoreId 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	CORECHECKSUM RO 0xB500B9															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CORECHECKSUM RO 0xB500B9								CORETYPEID RO 0x4							
fpga2sdram1_axi32_I_main_QosGenerator_Id_RevisionId 0x4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	FLEXNOCID RO 0x129FF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FLEXNOCID RO 0x129FF								USERID RO 0x0							
fpga2sdram1_axi32_I_main_QosGenerator_Priority 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	MARK RO 0x1	Reserved														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						P1 RW 0x2		Reserved						P0 RW 0x0	

Register Address Offset	Bit Fields															
fpga2sdram1_axi32_I_main_QosGenerator_Mode 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														MODE RW 0x3	
fpga2sdram1_axi32_I_main_QosGenerator_Bandwidth 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved					BANDWIDTH RW 0x280										
fpga2sdram1_axi32_I_main_QosGenerator_Saturation 0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						SATURATION RW 0x8									
fpga2sdram1_axi32_I_main_QosGenerator_ExtControl 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													INTCLKEN RW 0x0	EXTTHREN RW 0x0	SOCKETQOSEN RW 0x0

fpga2sdram1_axi32_I_main_QosGenerator_Id_CoreId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2sdram1_axi32_I_main_QosGenerator	0xFFD16800	0xFFD16800

Offset: 0x0

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CORECHECKSUM RO 0xB500B9															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CORECHECKSUM RO 0xB500B9								CORETYPEID RO 0x4							

fpga2sdram1_axi32_I_main_QosGenerator_Id_CoreId Fields

Bit	Name	Description	Access	Reset
31:8	CORECHECKSUM	Field containing a checksum of the parameters of the IP.	RO	0xB500B9
7:0	CORETYPEID	Field identifying the type of IP.	RO	0x4

fpga2sdram1_axi32_I_main_QosGenerator_Id_RevisionId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2sdram1_axi32_I_main_QosGenerator	0xFFD16800	0xFFD16804

Offset: 0x4

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLEXNOCID RO 0x129FF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLEXNOCID RO 0x129FF								USERID RO 0x0							

fpga2sdram1_axi32_I_main_QosGenerator_Id_RevisionId Fields

Bit	Name	Description	Access	Reset
31:8	FLEXNOCID	Field containing the build revision of the software used to generate the IP HDL code.	RO	0x129FF
7:0	USERID	Field containing a user defined value, not used anywhere inside the IP itself.	RO	0x0

fpga2sdram1_axi32_I_main_QosGenerator_Priority

Priority register.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2sdram1_axi32_I_main_QosGenerator	0xFFD16800	0xFFD16808

Offset: 0x8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MARK RO 0x1	Reserved														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						P1 RW 0x2		Reserved						P0 RW 0x0	

fpga2sdram1_axi32_I_main_QosGenerator_Priority Fields

Bit	Name	Description	Access	Reset
31	MARK	Backward compatibility marker when 0.	RO	0x1

Bit	Name	Description	Access	Reset
9:8	P1	In Programmable or Bandwidth Limiter mode, the priority level for read transactions. In Bandwidth regulator mode, the priority level when the used throughput is below the threshold. In Bandwidth Regulator mode, P1 should have a value equal or greater than P0.	RW	0x2
1:0	P0	In Programmable or Bandwidth Limiter mode, the priority level for write transactions. In Bandwidth Regulator mode, the priority level when the used throughput is above the threshold. In Bandwidth Regulator mode, P0 should have a value equal or lower than P1.	RW	0x0

fpga2sdram1_axi32_I_main_QosGenerator_Mode

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2sdram1_axi32_I_main_QosGenerator	0xFFD16800	0xFFD1680C

Offset: 0xC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														MODE RW 0x3	

fpga2sdram1_axi32_I_main_QosGenerator_Mode Fields

Bit	Name	Description	Access	Reset
1:0	MODE	0 = Programmable mode: a programmed priority is assigned to each read or write, 1 = Bandwidth Limiter Mode: a hard limit restricts throughput, 2 = Bypass mode: (<See SoC-specific QoS generator documentation>), 3 = Bandwidth Regulator mode: priority decreases when throughput exceeds a threshold.	RW	0x3

fpga2sdram1_axi32_I_main_QosGenerator_Bandwidth

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2sdram1_axi32_I_main_QosGenerator	0xFFD16800	0xFFD16810

Offset: 0x10

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					BANDWIDTH RW 0x280										

fpga2sdram1_axi32_I_main_QosGenerator_Bandwidth Fields

Bit	Name	Description	Access	Reset
10:0	BANDWIDTH	In Bandwidth Limiter or Bandwidth Regulator mode, the bandwidth threshold is in units of 1/256th bytes per cycle. The formula to calculate this value is: Bandwidth = Throughput (Mbsp) * 256 / Clock Frequency. For example, 80 MBps on a 250 MHz interface has the value 0x0052 (BW = 80Mbps * 256/ 250MHz = 81.92, which is rounded to 0x0052).	RW	0x280

fpga2sdram1_axi32_I_main_QosGenerator_Saturation

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2sdram1_axi32_I_main_QosGenerator	0xFFD16800	0xFFD16814

Offset: 0x14

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						SATURATION RW 0x8									

fpga2sdram1_axi32_I_main_QosGenerator_Saturation Fields

Bit	Name	Description	Access	Reset
9:0	SATURATION	In Bandwidth Limiter or Bandwidth Regulator mode, the maximum data count value, in units of 16 bytes. This determines the window of time over which bandwidth is measured. For example, to measure bandwidth within a 1000 cycle window on a 64-bit interface is value 0x1F4.	RW	0x8

fpga2sdram1_axi32_I_main_QosGenerator_ExtControl

External inputs control.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2sdram1_axi32_I_main_QosGenerator	0xFFD16800	0xFFD16818

Offset: 0x18

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													INTCLKEN RW 0x0	EXTTHREN RW 0x0	SOCKETQOSEN RW 0x0

fpga2sdram1_axi32_I_main_QosGenerator_ExtControl Fields

Bit	Name	Description	Access	Reset
2	INTCLKEN	n/a	RW	0x0
1	EXTTHREN	n/a	RW	0x0

Bit	Name	Description	Access	Reset
0	SOCKETQOSEN	n/a	RW	0x0

noc_mpu_fpga2sdram1_axi64_I_main_QosGenerator Address Map

Module Instance	Base Address	End Address
i_noc_mpu_m0_fpga2sdram1_axi64_I_main_QosGenerator	0xFFD16880	0xFFD168FF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
fpga2sdram1_axi64_I_main_QosGenerator_Id_CoreId on page 7-818	0x0	32	RO	0xE0E1EF04	
fpga2sdram1_axi64_I_main_QosGenerator_Id_RevisionId on page 7-819	0x4	32	RO	0x129FF00	
fpga2sdram1_axi64_I_main_QosGenerator_Priority on page 7-820	0x8	32	RW	0x80000200	Priority register.
fpga2sdram1_axi64_I_main_QosGenerator_Mode on page 7-821	0xC	32	RW	0x3	
fpga2sdram1_axi64_I_main_QosGenerator_Bandwidth on page 7-822	0x10	32	RW	0x780	
fpga2sdram1_axi64_I_main_QosGenerator_Saturation on page 7-823	0x14	32	RW	0x8	

Register	Offset	Width	Access	Reset Value	Description
fpga2sdram1_axi64_I_main_QosGenerator_ExtControl on page 7-824	0x18	32	RW	0x0	External inputs control.

noc_mpu_fpga2sdram1_axi64_I_main_QosGenerator Summary

Base Address: 0xFFD16880

Register Address Offset	Bit Fields															
i_noc_mpu_m0_fpga2sdram1_axi64_I_main_QosGenerator																
fpga2sdram1_axi64_I_main_QosGenerator_Id_CoreId 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	CORECHECKSUM RO 0xE0E1EF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CORECHECKSUM RO 0xE0E1EF								CORETYPEID RO 0x4							
fpga2sdram1_axi64_I_main_QosGenerator_Id_RevisionId 0x4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	FLEXNOCID RO 0x129FF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FLEXNOCID RO 0x129FF								USERID RO 0x0							
fpga2sdram1_axi64_I_main_QosGenerator_Priority 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	MARK RO 0x1	Reserved														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						P1 RW 0x2		Reserved						P0 RW 0x0	

Register Address Offset	Bit Fields															
fpga2sdram1_axi64_I_main_QosGenerator_Mode 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														MODE RW 0x3	
fpga2sdram1_axi64_I_main_QosGenerator_Bandwidth 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				BANDWIDTH RW 0x780											
fpga2sdram1_axi64_I_main_QosGenerator_Saturation 0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						SATURATION RW 0x8									
fpga2sdram1_axi64_I_main_QosGenerator_ExtControl 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													INTCLKEN RW 0x0	EXTTHREN RW 0x0	SOCKETQOSEN RW 0x0

fpga2sdram1_axi64_I_main_QosGenerator_Id_CoreId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2sdram1_axi64_I_main_QosGenerator	0xFFD16880	0xFFD16880

Offset: 0x0

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CORECHECKSUM RO 0xE0E1EF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CORECHECKSUM RO 0xE0E1EF								CORETYPEID RO 0x4							

fpga2sdram1_axi64_l_main_QosGenerator_Id_CoreId Fields

Bit	Name	Description	Access	Reset
31:8	CORECHECKSUM	Field containing a checksum of the parameters of the IP.	RO	0xE0E1EF
7:0	CORETYPEID	Field identifying the type of IP.	RO	0x4

fpga2sdram1_axi64_l_main_QosGenerator_Id_RevisionId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2sdram1_axi64_l_main_QosGenerator	0xFFD16880	0xFFD16884

Offset: 0x4

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLEXNOCID RO 0x129FF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLEXNOCID RO 0x129FF								USERID RO 0x0							

fpga2sdram1_axi64_I_main_QosGenerator_Id_RevisionId Fields

Bit	Name	Description	Access	Reset
31:8	FLEXNOCID	Field containing the build revision of the software used to generate the IP HDL code.	RO	0x129FF
7:0	USERID	Field containing a user defined value, not used anywhere inside the IP itself.	RO	0x0

fpga2sdram1_axi64_I_main_QosGenerator_Priority

Priority register.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2sdram1_axi64_I_main_QosGenerator	0xFFD16880	0xFFD16888

Offset: 0x8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MARK RO 0x1	Reserved														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						P1 RW 0x2		Reserved						P0 RW 0x0	

fpga2sdram1_axi64_I_main_QosGenerator_Priority Fields

Bit	Name	Description	Access	Reset
31	MARK	Backward compatibility marker when 0.	RO	0x1

Bit	Name	Description	Access	Reset
9:8	P1	In Programmable or Bandwidth Limiter mode, the priority level for read transactions. In Bandwidth regulator mode, the priority level when the used throughput is below the threshold. In Bandwidth Regulator mode, P1 should have a value equal or greater than P0.	RW	0x2
1:0	P0	In Programmable or Bandwidth Limiter mode, the priority level for write transactions. In Bandwidth Regulator mode, the priority level when the used throughput is above the threshold. In Bandwidth Regulator mode, P0 should have a value equal or lower than P1.	RW	0x0

fpga2sdram1_axi64_I_main_QosGenerator_Mode

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2sdram1_axi64_I_main_QosGenerator	0xFFD16880	0xFFD1688C

Offset: 0xC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														MODE RW 0x3	

fpga2sdram1_axi64_I_main_QosGenerator_Mode Fields

Bit	Name	Description	Access	Reset
1:0	MODE	0 = Programmable mode: a programmed priority is assigned to each read or write, 1 = Bandwidth Limiter Mode: a hard limit restricts throughput, 2 = Bypass mode: (<See SoC-specific QoS generator documentation>), 3 = Bandwidth Regulator mode: priority decreases when throughput exceeds a threshold.	RW	0x3

fpga2sdram1_axi64_I_main_QosGenerator_Bandwidth

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2sdram1_axi64_I_main_QosGenerator	0xFFD16880	0xFFD16890

Offset: 0x10

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				BANDWIDTH RW 0x780											

fpga2sdram1_axi64_I_main_QosGenerator_Bandwidth Fields

Bit	Name	Description	Access	Reset
11:0	BANDWIDTH	In Bandwidth Limiter or Bandwidth Regulator mode, the bandwidth threshold is in units of 1/256th bytes per cycle. The formula to calculate this value is: Bandwidth = Throughput (Mbsp) * 256 / Clock Frequency. For example, 80 MBps on a 250 MHz interface has the value 0x0052 (BW = 80Mbps * 256/ 250MHz = 81.92, which is rounded to 0x0052).	RW	0x780

fpga2sdram1_axi64_I_main_QosGenerator_Saturation

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2sdram1_axi64_I_main_QosGenerator	0xFFD16880	0xFFD16894

Offset: 0x14

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						SATURATION RW 0x8									

fpga2sdram1_axi64_I_main_QosGenerator_Saturation Fields

Bit	Name	Description	Access	Reset
9:0	SATURATION	In Bandwidth Limiter or Bandwidth Regulator mode, the maximum data count value, in units of 16 bytes. This determines the window of time over which bandwidth is measured. For example, to measure bandwidth within a 1000 cycle window on a 64-bit interface is value 0x1F4.	RW	0x8

fpga2sdram1_axi64_I_main_QosGenerator_ExtControl

External inputs control.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2sdram1_axi64_I_main_QosGenerator	0xFFD16880	0xFFD16898

Offset: 0x18

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													INTCLKEN RW 0x0	EXTTHREN RW 0x0	SOCKETQOSEN RW 0x0

fpga2sdram1_axi64_I_main_QosGenerator_ExtControl Fields

Bit	Name	Description	Access	Reset
2	INTCLKEN	n/a	RW	0x0
1	EXTTHREN	n/a	RW	0x0

Bit	Name	Description	Access	Reset
0	SOCKETQOSEN	n/a	RW	0x0

noc_mpu_fpga2sdram2_axi32_I_main_QosGenerator Address Map

Module Instance	Base Address	End Address
i_noc_mpu_m0_fpga2sdram2_axi32_I_main_QosGenerator	0xFFD16900	0xFFD1697F

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
fpga2sdram2_axi32_I_main_QosGenerator_Id_CoreId on page 7-827	0x0	32	RO	0x653D2204	
fpga2sdram2_axi32_I_main_QosGenerator_Id_RevisionId on page 7-828	0x4	32	RO	0x129FF00	
fpga2sdram2_axi32_I_main_QosGenerator_Priority on page 7-829	0x8	32	RW	0x80000200	Priority register.
fpga2sdram2_axi32_I_main_QosGenerator_Mode on page 7-830	0xC	32	RW	0x3	
fpga2sdram2_axi32_I_main_QosGenerator_Bandwidth on page 7-831	0x10	32	RW	0x280	
fpga2sdram2_axi32_I_main_QosGenerator_Saturation on page 7-832	0x14	32	RW	0x8	

Register	Offset	Width	Access	Reset Value	Description
fpga2sdram2_axi32_I_main_QosGenerator_ExtControl on page 7-833	0x18	32	RW	0x0	External inputs control.

noc_mpu_fpga2sdram2_axi32_I_main_QosGenerator Summary

Base Address: 0xFFD16900

Register Address Offset	Bit Fields															
i_noc_mpu_m0_fpga2sdram2_axi32_I_main_QosGenerator																
fpga2sdram2_axi32_I_main_QosGenerator_Id_CoreId 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	CORECHECKSUM RO 0x653D22															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CORECHECKSUM RO 0x653D22								CORETYPEID RO 0x4							
fpga2sdram2_axi32_I_main_QosGenerator_Id_RevisionId 0x4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	FLEXNOCID RO 0x129FF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FLEXNOCID RO 0x129FF								USERID RO 0x0							
fpga2sdram2_axi32_I_main_QosGenerator_Priority 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	MARK RO 0x1	Reserved														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						P1 RW 0x2		Reserved						P0 RW 0x0	

Register Address Offset	Bit Fields															
fpga2sdram2_axi32_I_main_QosGenerator_Mode 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														MODE RW 0x3	
fpga2sdram2_axi32_I_main_QosGenerator_Bandwidth 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved					BANDWIDTH RW 0x280										
fpga2sdram2_axi32_I_main_QosGenerator_Saturation 0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						SATURATION RW 0x8									
fpga2sdram2_axi32_I_main_QosGenerator_ExtControl 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													INTCLKEN RW 0x0	EXTTHREN RW 0x0	SOCKETQOSEN RW 0x0

fpga2sdram2_axi32_I_main_QosGenerator_Id_CoreId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2sdram2_axi32_I_main_QosGenerator	0xFFD16900	0xFFD16900

Offset: 0x0

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CORECHECKSUM RO 0x653D22															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CORECHECKSUM RO 0x653D22								CORETYPEID RO 0x4							

fpga2sdram2_axi32_I_main_QosGenerator_Id_CoreId Fields

Bit	Name	Description	Access	Reset
31:8	CORECHECKSUM	Field containing a checksum of the parameters of the IP.	RO	0x653D22
7:0	CORETYPEID	Field identifying the type of IP.	RO	0x4

fpga2sdram2_axi32_I_main_QosGenerator_Id_RevisionId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2sdram2_axi32_I_main_QosGenerator	0xFFD16900	0xFFD16904

Offset: 0x4

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLEXNOCID RO 0x129FF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLEXNOCID RO 0x129FF								USERID RO 0x0							

fpga2sdram2_axi32_I_main_QosGenerator_Id_RevisionId Fields

Bit	Name	Description	Access	Reset
31:8	FLEXNOCID	Field containing the build revision of the software used to generate the IP HDL code.	RO	0x129FF
7:0	USERID	Field containing a user defined value, not used anywhere inside the IP itself.	RO	0x0

fpga2sdram2_axi32_I_main_QosGenerator_Priority

Priority register.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2sdram2_axi32_I_main_QosGenerator	0xFFD16900	0xFFD16908

Offset: 0x8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MARK RO 0x1	Reserved														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						P1 RW 0x2		Reserved						P0 RW 0x0	

fpga2sdram2_axi32_I_main_QosGenerator_Priority Fields

Bit	Name	Description	Access	Reset
31	MARK	Backward compatibility marker when 0.	RO	0x1

Bit	Name	Description	Access	Reset
9:8	P1	In Programmable or Bandwidth Limiter mode, the priority level for read transactions. In Bandwidth regulator mode, the priority level when the used throughput is below the threshold. In Bandwidth Regulator mode, P1 should have a value equal or greater than P0.	RW	0x2
1:0	P0	In Programmable or Bandwidth Limiter mode, the priority level for write transactions. In Bandwidth Regulator mode, the priority level when the used throughput is above the threshold. In Bandwidth Regulator mode, P0 should have a value equal or lower than P1.	RW	0x0

fpga2sdram2_axi32_I_main_QosGenerator_Mode

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2sdram2_axi32_I_main_QosGenerator	0xFFD16900	0xFFD1690C

Offset: 0xC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														MODE RW 0x3	

fpga2sdram2_axi32_I_main_QosGenerator_Mode Fields

Bit	Name	Description	Access	Reset
1:0	MODE	0 = Programmable mode: a programmed priority is assigned to each read or write, 1 = Bandwidth Limiter Mode: a hard limit restricts throughput, 2 = Bypass mode: (<See SoC-specific QoS generator documentation>), 3 = Bandwidth Regulator mode: priority decreases when throughput exceeds a threshold.	RW	0x3

fpga2sdram2_axi32_I_main_QosGenerator_Bandwidth

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2sdram2_axi32_I_main_QosGenerator	0xFFD16900	0xFFD16910

Offset: 0x10

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					BANDWIDTH RW 0x280										

fpga2sdram2_axi32_I_main_QosGenerator_Bandwidth Fields

Bit	Name	Description	Access	Reset
10:0	BANDWIDTH	In Bandwidth Limiter or Bandwidth Regulator mode, the bandwidth threshold is in units of 1/256th bytes per cycle. The formula to calculate this value is: Bandwidth = Throughput (Mbsp) * 256 / Clock Frequency. For example, 80 MBps on a 250 MHz interface has the value 0x0052 (BW = 80Mbps * 256/ 250MHz = 81.92, which is rounded to 0x0052).	RW	0x280

fpga2sdram2_axi32_I_main_QosGenerator_Saturation

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2sdram2_axi32_I_main_QosGenerator	0xFFD16900	0xFFD16914

Offset: 0x14

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						SATURATION RW 0x8									

fpga2sdram2_axi32_I_main_QosGenerator_Saturation Fields

Bit	Name	Description	Access	Reset
9:0	SATURATION	In Bandwidth Limiter or Bandwidth Regulator mode, the maximum data count value, in units of 16 bytes. This determines the window of time over which bandwidth is measured. For example, to measure bandwidth within a 1000 cycle window on a 64-bit interface is value 0x1F4.	RW	0x8

fpga2sdram2_axi32_I_main_QosGenerator_ExtControl

External inputs control.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2sdram2_axi32_I_main_QosGenerator	0xFFD16900	0xFFD16918

Offset: 0x18

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													INTCLKEN RW 0x0	EXTTHREN RW 0x0	SOCKETQOSEN RW 0x0

fpga2sdram2_axi32_I_main_QosGenerator_ExtControl Fields

Bit	Name	Description	Access	Reset
2	INTCLKEN	n/a	RW	0x0
1	EXTTHREN	n/a	RW	0x0

Bit	Name	Description	Access	Reset
0	SOCKETQOSEN	n/a	RW	0x0

noc_mpu_fpga2sdram2_axi64_I_main_QosGenerator Address Map

Module Instance	Base Address	End Address
i_noc_mpu_m0_fpga2sdram2_axi64_I_main_QosGenerator	0xFFD16980	0xFFD16FFF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
fpga2sdram2_axi64_I_main_QosGenerator_Id_CoreId on page 7-836	0x0	32	RO	0xD551CF04	
fpga2sdram2_axi64_I_main_QosGenerator_Id_RevisionId on page 7-837	0x4	32	RO	0x129FF00	
fpga2sdram2_axi64_I_main_QosGenerator_Priority on page 7-838	0x8	32	RW	0x80000200	Priority register.
fpga2sdram2_axi64_I_main_QosGenerator_Mode on page 7-839	0xC	32	RW	0x3	
fpga2sdram2_axi64_I_main_QosGenerator_Bandwidth on page 7-840	0x10	32	RW	0x780	
fpga2sdram2_axi64_I_main_QosGenerator_Saturation on page 7-841	0x14	32	RW	0x8	

Register	Offset	Width	Access	Reset Value	Description
fpga2sdram2_axi64_I_main_QosGenerator_ExtControl on page 7-842	0x18	32	RW	0x0	External inputs control.

noc_mpu_fpga2sdram2_axi64_I_main_QosGenerator Summary

Base Address: 0xFFD16980

Register Address Offset	Bit Fields															
i_noc_mpu_m0_fpga2sdram2_axi64_I_main_QosGenerator																
fpga2sdram2_axi64_I_main_QosGenerator_Id_CoreId 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	CORECHECKSUM RO 0xD551CF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CORECHECKSUM RO 0xD551CF								CORETYPEID RO 0x4							
fpga2sdram2_axi64_I_main_QosGenerator_Id_RevisionId 0x4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	FLEXNOCID RO 0x129FF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FLEXNOCID RO 0x129FF								USERID RO 0x0							
fpga2sdram2_axi64_I_main_QosGenerator_Priority 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	MARK RO 0x1	Reserved														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						P1 RW 0x2		Reserved						P0 RW 0x0	

Register Address Offset	Bit Fields															
fpga2sdram2_axi64_I_main_QosGenerator_Mode 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														MODE RW 0x3	
fpga2sdram2_axi64_I_main_QosGenerator_Bandwidth 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				BANDWIDTH RW 0x780											
fpga2sdram2_axi64_I_main_QosGenerator_Saturation 0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						SATURATION RW 0x8									
fpga2sdram2_axi64_I_main_QosGenerator_ExtControl 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													INTCLKEN RW 0x0	EXTTHREN RW 0x0	SOCKETQOSEN RW 0x0

fpga2sdram2_axi64_I_main_QosGenerator_Id_CoreId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2sdram2_axi64_I_main_QosGenerator	0xFFD16980	0xFFD16980

Offset: 0x0

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CORECHECKSUM RO 0xD551CF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CORECHECKSUM RO 0xD551CF								CORETYPEID RO 0x4							

fpga2sdram2_axi64_I_main_QosGenerator_Id_CoreId Fields

Bit	Name	Description	Access	Reset
31:8	CORECHECKSUM	Field containing a checksum of the parameters of the IP.	RO	0xD551CF
7:0	CORETYPEID	Field identifying the type of IP.	RO	0x4

fpga2sdram2_axi64_I_main_QosGenerator_Id_RevisionId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2sdram2_axi64_I_main_QosGenerator	0xFFD16980	0xFFD16984

Offset: 0x4

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLEXNOCID RO 0x129FF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLEXNOCID RO 0x129FF								USERID RO 0x0							

fpga2sdram2_axi64_I_main_QosGenerator_Id_RevisionId Fields

Bit	Name	Description	Access	Reset
31:8	FLEXNOCID	Field containing the build revision of the software used to generate the IP HDL code.	RO	0x129FF
7:0	USERID	Field containing a user defined value, not used anywhere inside the IP itself.	RO	0x0

fpga2sdram2_axi64_I_main_QosGenerator_Priority

Priority register.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2sdram2_axi64_I_main_QosGenerator	0xFFD16980	0xFFD16988

Offset: 0x8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MARK RO 0x1	Reserved														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						P1 RW 0x2		Reserved						P0 RW 0x0	

fpga2sdram2_axi64_I_main_QosGenerator_Priority Fields

Bit	Name	Description	Access	Reset
31	MARK	Backward compatibility marker when 0.	RO	0x1

Bit	Name	Description	Access	Reset
9:8	P1	In Programmable or Bandwidth Limiter mode, the priority level for read transactions. In Bandwidth regulator mode, the priority level when the used throughput is below the threshold. In Bandwidth Regulator mode, P1 should have a value equal or greater than P0.	RW	0x2
1:0	P0	In Programmable or Bandwidth Limiter mode, the priority level for write transactions. In Bandwidth Regulator mode, the priority level when the used throughput is above the threshold. In Bandwidth Regulator mode, P0 should have a value equal or lower than P1.	RW	0x0

fpga2sdram2_axi64_I_main_QosGenerator_Mode

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2sdram2_axi64_I_main_QosGenerator	0xFFD16980	0xFFD1698C

Offset: 0xC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														MODE RW 0x3	

fpga2sdram2_axi64_I_main_QosGenerator_Mode Fields

Bit	Name	Description	Access	Reset
1:0	MODE	0 = Programmable mode: a programmed priority is assigned to each read or write, 1 = Bandwidth Limiter Mode: a hard limit restricts throughput, 2 = Bypass mode: (<See SoC-specific QoS generator documentation>), 3 = Bandwidth Regulator mode: priority decreases when throughput exceeds a threshold.	RW	0x3

fpga2sdram2_axi64_I_main_QosGenerator_Bandwidth

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2sdram2_axi64_I_main_QosGenerator	0xFFD16980	0xFFD16990

Offset: 0x10

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				BANDWIDTH RW 0x780											

fpga2sdram2_axi64_I_main_QosGenerator_Bandwidth Fields

Bit	Name	Description	Access	Reset
11:0	BANDWIDTH	In Bandwidth Limiter or Bandwidth Regulator mode, the bandwidth threshold is in units of 1/256th bytes per cycle. The formula to calculate this value is: Bandwidth = Throughput (Mbsp) * 256 / Clock Frequency. For example, 80 MBps on a 250 MHz interface has the value 0x0052 (BW = 80Mbps * 256/ 250MHz = 81.92, which is rounded to 0x0052).	RW	0x780

fpga2sdram2_axi64_I_main_QosGenerator_Saturation

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2sdram2_axi64_I_main_QosGenerator	0xFFD16980	0xFFD16994

Offset: 0x14

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						SATURATION RW 0x8									

fpga2sdram2_axi64_I_main_QosGenerator_Saturation Fields

Bit	Name	Description	Access	Reset
9:0	SATURATION	In Bandwidth Limiter or Bandwidth Regulator mode, the maximum data count value, in units of 16 bytes. This determines the window of time over which bandwidth is measured. For example, to measure bandwidth within a 1000 cycle window on a 64-bit interface is value 0x1F4.	RW	0x8

fpga2sdram2_axi64_I_main_QosGenerator_ExtControl

External inputs control.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2sdram2_axi64_I_main_QosGenerator	0xFFD16980	0xFFD16998

Offset: 0x18

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													INTCLKEN RW 0x0	EXTTHREN RW 0x0	SOCKETQOSEN RW 0x0

fpga2sdram2_axi64_I_main_QosGenerator_ExtControl Fields

Bit	Name	Description	Access	Reset
2	INTCLKEN	n/a	RW	0x0
1	EXTTHREN	n/a	RW	0x0

Bit	Name	Description	Access	Reset
0	SOCKETQOSEN	n/a	RW	0x0

noc_mpu_fpga2sdram2_axi128_I_main_QosGenerator Address Map

Module Instance	Base Address	End Address
i_noc_mpu_m0_fpga2sdram2_axi128_I_main_QosGenerator	0xFFD17000	0xFFD1707F

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
fpga2sdram2_axi128_I_main_QosGenerator_Id_CoreId on page 7-845	0x0	32	RO	0xD4237F04	
fpga2sdram2_axi128_I_main_QosGenerator_Id_RevisionId on page 7-846	0x4	32	RO	0x129FF00	
fpga2sdram2_axi128_I_main_QosGenerator_Priority on page 7-847	0x8	32	RW	0x80000200	Priority register.
fpga2sdram2_axi128_I_main_QosGenerator_Mode on page 7-848	0xC	32	RW	0x3	
fpga2sdram2_axi128_I_main_QosGenerator_Bandwidth on page 7-849	0x10	32	RW	0xC80	
fpga2sdram2_axi128_I_main_QosGenerator_Saturation on page 7-850	0x14	32	RW	0x8	

Register	Offset	Width	Access	Reset Value	Description
fpga2sdram2_axi128_I_main_QosGenerator_ExtControl on page 7-851	0x18	32	RW	0x0	External inputs control.

noc_mpu_fpga2sdram2_axi128_I_main_QosGenerator Summary

Base Address: 0xFFD17000

Register Address Offset	Bit Fields															
i_noc_mpu_m0_fpga2sdram2_axi128_I_main_QosGenerator																
fpga2sdram2_axi128_I_main_QosGenerator_Id_CoreId 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	CORECHECKSUM RO 0xD4237F															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CORECHECKSUM RO 0xD4237F								CORETYPEID RO 0x4							
fpga2sdram2_axi128_I_main_QosGenerator_Id_RevisionId 0x4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	FLEXNOCID RO 0x129FF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FLEXNOCID RO 0x129FF								USERID RO 0x0							
fpga2sdram2_axi128_I_main_QosGenerator_Priority 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	MARK RO 0x1	Reserved														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						P1 RW 0x2		Reserved						P0 RW 0x0	

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
fpga2sdram2_axi128_I_main_QosGenerator_Mode 0xC	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														MODE RW 0x3	
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
fpga2sdram2_axi128_I_main_QosGenerator_Bandwidth 0x10	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved			BANDWIDTH RW 0xC80												
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
fpga2sdram2_axi128_I_main_QosGenerator_Saturation 0x14	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						SATURATION RW 0x8									
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
fpga2sdram2_axi128_I_main_QosGenerator_ExtControl 0x18	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													INTCLKEN RW 0x0	EXTTHREN RW 0x0	SOCKETQOSEN RW 0x0

fpga2sdram2_axi128_I_main_QosGenerator_Id_CoreId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2sdram2_axi128_I_main_QosGenerator	0xFFD17000	0xFFD17000

Offset: 0x0

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CORECHECKSUM RO 0xD4237F															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CORECHECKSUM RO 0xD4237F								CORETYPEID RO 0x4							

fpga2sdram2_axi128_I_main_QosGenerator_Id_CoreId Fields

Bit	Name	Description	Access	Reset
31:8	CORECHECKSUM	Field containing a checksum of the parameters of the IP.	RO	0xD4237F
7:0	CORETYPEID	Field identifying the type of IP.	RO	0x4

fpga2sdram2_axi128_I_main_QosGenerator_Id_RevisionId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2sdram2_axi128_I_main_QosGenerator	0xFFD17000	0xFFD17004

Offset: 0x4

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLEXNOCID RO 0x129FF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLEXNOCID RO 0x129FF								USERID RO 0x0							

fpga2sdram2_axi128_I_main_QosGenerator_Id_RevisionId Fields

Bit	Name	Description	Access	Reset
31:8	FLEXNOCID	Field containing the build revision of the software used to generate the IP HDL code.	RO	0x129FF
7:0	USERID	Field containing a user defined value, not used anywhere inside the IP itself.	RO	0x0

fpga2sdram2_axi128_I_main_QosGenerator_Priority

Priority register.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ fpga2sdram2_axi128_I_ main_QosGenerator	0xFFD17000	0xFFD17008

Offset: 0x8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MARK RO 0x1	Reserved														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						P1 RW 0x2		Reserved						P0 RW 0x0	

fpga2sdram2_axi128_I_main_QosGenerator_Priority Fields

Bit	Name	Description	Access	Reset
31	MARK	Backward compatibility marker when 0.	RO	0x1

Bit	Name	Description	Access	Reset
9:8	P1	In Programmable or Bandwidth Limiter mode, the priority level for read transactions. In Bandwidth regulator mode, the priority level when the used throughput is below the threshold. In Bandwidth Regulator mode, P1 should have a value equal or greater than P0.	RW	0x2
1:0	P0	In Programmable or Bandwidth Limiter mode, the priority level for write transactions. In Bandwidth Regulator mode, the priority level when the used throughput is above the threshold. In Bandwidth Regulator mode, P0 should have a value equal or lower than P1.	RW	0x0

fpga2sdram2_axi128_I_main_QosGenerator_Mode

Module Instance	Base Address	Register Address
i_noc_mpu_m0_ fpga2sdram2_axi128_I_ main_QosGenerator	0xFFD17000	0xFFD1700C

Offset: 0xC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														MODE RW 0x3	

fpga2sdram2_axi128_I_main_QosGenerator_Mode Fields

Bit	Name	Description	Access	Reset
1:0	MODE	0 = Programmable mode: a programmed priority is assigned to each read or write, 1 = Bandwidth Limiter Mode: a hard limit restricts throughput, 2 = Bypass mode: (<See SoC-specific QoS generator documentation>), 3 = Bandwidth Regulator mode: priority decreases when throughput exceeds a threshold.	RW	0x3

fpga2sdram2_axi128_I_main_QosGenerator_Bandwidth

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2sdram2_axi128_I_main_QosGenerator	0xFFD17000	0xFFD17010

Offset: 0x10

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			BANDWIDTH RW 0xC80												

fpga2sdram2_axi128_I_main_QosGenerator_Bandwidth Fields

Bit	Name	Description	Access	Reset
12:0	BANDWIDTH	In Bandwidth Limiter or Bandwidth Regulator mode, the bandwidth threshold is in units of 1/256th bytes per cycle. The formula to calculate this value is: Bandwidth = Throughput (Mbsp) * 256 / Clock Frequency. For example, 80 MBps on a 250 MHz interface has the value 0x0052 (BW = 80Mbps * 256/ 250MHz = 81.92, which is rounded to 0x0052).	RW	0xC80

fpga2sdram2_axi128_I_main_QosGenerator_Saturation

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2sdram2_axi128_I_main_QosGenerator	0xFFD17000	0xFFD17014

Offset: 0x14

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						SATURATION RW 0x8									

fpga2sdram2_axi128_I_main_QosGenerator_Saturation Fields

Bit	Name	Description	Access	Reset
9:0	SATURATION	In Bandwidth Limiter or Bandwidth Regulator mode, the maximum data count value, in units of 16 bytes. This determines the window of time over which bandwidth is measured. For example, to measure bandwidth within a 1000 cycle window on a 64-bit interface is value 0x1F4.	RW	0x8

fpga2sdram2_axi128_I_main_QosGenerator_ExtControl

External inputs control.

Module Instance	Base Address	Register Address
i_noc_mpu_m0_fpga2sdram2_axi128_I_main_QosGenerator	0xFFD17000	0xFFD17018

Offset: 0x18

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													INTCLKEN RW 0x0	EXTTHREN RW 0x0	SOCKETQOSEN RW 0x0

fpga2sdram2_axi128_I_main_QosGenerator_ExtControl Fields

Bit	Name	Description	Access	Reset
2	INTCLKEN	n/a	RW	0x0
1	EXTTHREN	n/a	RW	0x0

Bit	Name	Description	Access	Reset
0	SOCKETQOSEN	n/a	RW	0x0

noc_mpu_emac0_m_I_main_TransactionStatFilter Address Map

Module Instance	Base Address	End Address
i_noc_mpu_m0_emac0_m_I_main_TransactionStatFilter	0xFFD17080	0xFFD9FFFF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
emac0_m_I_main_TransactionStat-Filter_Id_CoreId on page 7-855	0x0	32	RO	0xC284D309	
emac0_m_I_main_TransactionStat-Filter_Id_RevisionId on page 7-856	0x4	32	RO	0x129FF00	
emac0_m_I_main_TransactionStatFilter_Mode on page 7-857	0x8	32	RW	0x0	
emac0_m_I_main_TransactionStat-Filter_AddrBase_Low on page 7-858	0xC	32	RW	0x0	
emac0_m_I_main_TransactionStat-Filter_AddrBase_High on page 7-858	0x10	32	RW	0x0	
emac0_m_I_main_TransactionStat-Filter_AddrWindowSize on page 7-859	0x14	32	RW	0x0	

Register	Offset	Width	Access	Reset Value	Description
emac0_m_l_main_TransactionStat-Filter_Opcode on page 7-860	0x20	32	RW	0x0	This register selects candidate packets based on packet opcodes. (0 disables the filter):
emac0_m_l_main_TransactionStat-Filter_UserBase on page 7-861	0x24	32	RW	0x0	
emac0_m_l_main_TransactionStat-Filter_UserMask on page 7-861	0x28	32	RW	0x0	
emac0_m_l_main_TransactionStat-Filter_SecurityBase on page 7-862	0x2C	32	RW	0x0	
emac0_m_l_main_TransactionStat-Filter_SecurityMask on page 7-863	0x30	32	RW	0x0	

noc_mpu_emac0_m_l_main_TransactionStatFilter Summary

Base Address: 0xFFD17080

Register Address Offset	Bit Fields															
i_noc_mpu_m0_emac0_m_l_main_Transac-tionStat-Filter																
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
emac0_m_l_main_TransactionStat-Filter_Id_CoreId	CORECHECKSUM RO 0xC284D3															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x0	CORECHECKSUM RO 0xC284D3								CORETYPEID RO 0x9							

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
emac0_m_l_main_TransactionStat-Filter_Id_RevisionId 0x4	FLEXNOCID RO 0x129FF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FLEXNOCID RO 0x129FF								USERID RO 0x0							
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
emac0_m_l_main_TransactionStat-Filter_Mode 0x8	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															MODE RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
emac0_m_l_main_TransactionStat-Filter_AddrBase_Low 0xC	ADDRBASE_LOW RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ADDRBASE_LOW RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
emac0_m_l_main_TransactionStat-Filter_AddrBase_High 0x10	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															ADDRBASE_HIGH RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
emac0_m_l_main_TransactionStat-Filter_AddrWindowSize 0x14	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved										ADDRWINDOWSIZE RW 0x0					
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
emac0_m_l_main_TransactionStat-Filter_Opcode 0x20	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														WREN RW 0x0	RDEN RW 0x0

Register Address Offset	Bit Fields															
emac0_m_I_main_TransactionStat-Filter_UserBase 0x24	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved					USERBASE RW 0x0										
emac0_m_I_main_TransactionStat-Filter_UserMask 0x28	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved					USERMASK RW 0x0										
emac0_m_I_main_TransactionStat-Filter_SecurityBase 0x2C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													SECURITYBASE RW 0x0		
emac0_m_I_main_TransactionStat-Filter_SecurityMask 0x30	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													SECURITYMASK RW 0x0		

emac0_m_l_main_TransactionStatFilter_Id_CoreId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_emac0_m_I_main_TransactionStatFilter	0xFFD17080	0xFFD17080

Offset: 0x0

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CORECHECKSUM RO 0xC284D3															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CORECHECKSUM RO 0xC284D3								CORETYPEID RO 0x9							

emac0_m_I_main_TransactionStatFilter_Id_CoreId Fields

Bit	Name	Description	Access	Reset
31:8	CORECHECKSUM	Field containing a checksum of the parameters of the IP.	RO	0xC284D3
7:0	CORETYPEID	Field identifying the type of IP.	RO	0x9

emac0_m_I_main_TransactionStatFilter_Id_RevisionId

Module Instance	Base Address	Register Address
i_noc_mpu_m0_emac0_m_I_main_TransactionStatFilter	0xFFD17080	0xFFD17084

Offset: 0x4

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLEXNOCID RO 0x129FF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLEXNOCID RO 0x129FF								USERID RO 0x0							

emac0_m_l_main_TransactionStatFilter_Id_RevisionId Fields

Bit	Name	Description	Access	Reset
31:8	FLEXNOCID	Field containing the build revision of the software used to generate the IP HDL code.	RO	0x129FF
7:0	USERID	Field containing a user defined value, not used anywhere inside the IP itself.	RO	0x0

emac0_m_l_main_TransactionStatFilter_Mode

Module Instance	Base Address	Register Address
i_noc_mpu_m0_emac0_m_I_main_TransactionStatFilter	0xFFD17080	0xFFD17088

Offset: 0x8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															MODE RW 0x0

emac0_m_l_main_TransactionStatFilter_Mode Fields

Bit	Name	Description	Access	Reset
0	MODE	Register Mode is a 1-bit register that sets the filtering mode as follows: handshake Mode = 0 or latency Mode = 1.	RW	0x0

emac0_m_I_main_TransactionStatFilter_AddrBase_Low

Module Instance	Base Address	Register Address
i_noc_mpu_m0_emac0_m_I_main_TransactionStatFilter	0xFFD17080	0xFFD1708C

Offset: 0xC

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDRBASE_LOW RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRBASE_LOW RW 0x0															

emac0_m_I_main_TransactionStatFilter_AddrBase_Low Fields

Bit	Name	Description	Access	Reset
31:0	ADDRBASE_LOW	Address base LSB register.	RW	0x0

emac0_m_I_main_TransactionStatFilter_AddrBase_High

Module Instance	Base Address	Register Address
i_noc_mpu_m0_emac0_m_I_main_TransactionStatFilter	0xFFD17080	0xFFD17090

Offset: 0x10

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															ADDRBASE_HIGH RW 0x0

emac0_m_l_main_TransactionStatFilter_AddrBase_High Fields

Bit	Name	Description	Access	Reset
0	ADDRBASE_HIGH	Address base MSB register.	RW	0x0

emac0_m_l_main_TransactionStatFilter_AddrWindowSize

Module Instance	Base Address	Register Address
i_noc_mpu_m0_emac0_m_I_main_TransactionStatFilter	0xFFD17080	0xFFD17094

Offset: 0x14

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										ADDRWINDOWSIZE RW 0x0					

emac0_m_l_main_TransactionStatFilter_AddrWindowSize Fields

Bit	Name	Description	Access	Reset
5:0	ADDRWINDOWSIZE	Register AddrWindowSize contains the encoded address mask used to filter packets: the effective Mask value is equal to ~(2**AddrWindow-Size - 1). A packet is a candidate when ReqInfo.Addr & AddrMask = AddrBase & AddrMask.	RW	0x0

emac0_m_l_main_TransactionStatFilter_Opcode

This register selects candidate packets based on packet opcodes. (0 disables the filter):

Module Instance	Base Address	Register Address
i_noc_mpu_m0_emac0_m_l_main_TransactionStatFilter	0xFFD17080	0xFFD170A0

Offset: 0x20

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														WREN RW 0x0	RDEN RW 0x0

emac0_m_l_main_TransactionStatFilter_Opcode Fields

Bit	Name	Description	Access	Reset
1	WREN	When set to 1, selects WR requests.	RW	0x0

Bit	Name	Description	Access	Reset
0	RDEN	When set to 1, selects RD requests.	RW	0x0

emac0_m_l_main_TransactionStatFilter_UserBase

Module Instance	Base Address	Register Address
i_noc_mpu_m0_emac0_m_I_main_TransactionStatFilter	0xFFD17080	0xFFD170A4

Offset: 0x24

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					USERBASE RW 0x0										

emac0_m_l_main_TransactionStatFilter_UserBase Fields

Bit	Name	Description	Access	Reset
10:0	USERBASE	This register contains the User base used to filter requests.	RW	0x0

emac0_m_l_main_TransactionStatFilter_UserMask

Module Instance	Base Address	Register Address
i_noc_mpu_m0_emac0_m_I_main_TransactionStatFilter	0xFFD17080	0xFFD170A8

Offset: 0x28

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					USERMASK RW 0x0										

emac0_m_I_main_TransactionStatFilter_UserMask Fields

Bit	Name	Description	Access	Reset
10:0	USERMASK	This register contains the User mask used to filter requests.	RW	0x0

emac0_m_I_main_TransactionStatFilter_SecurityBase

Module Instance	Base Address	Register Address
i_noc_mpu_m0_emac0_m_I_main_TransactionStatFilter	0xFFD17080	0xFFD170AC

Offset: 0x2C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													SECURITYBASE RW 0x0		

emac0_m_l_main_TransactionStatFilter_SecurityBase Fields

Bit	Name	Description	Access	Reset
2:0	SECURITYBASE	This register contains the Security base used to filter requests.	RW	0x0

emac0_m_l_main_TransactionStatFilter_SecurityMask

Module Instance	Base Address	Register Address
i_noc_mpu_m0_emac0_m_I_main_TransactionStatFilter	0xFFD17080	0xFFD170B0

Offset: 0x30

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													SECURITYMASK RW 0x0		

emac0_m_l_main_TransactionStatFilter_SecurityMask Fields

Bit	Name	Description	Access	Reset
2:0	SECURITYMASK	This register contains the Security mask used to filter requests.	RW	0x0

Document Revision History

Table 7-22: Document Revision History

Date	Version	Changes
October 2016	2016.10.28	• "System Interconnect Slave Interfaces" table: Corrected acceptance values for Lightweight HPS-to-FPGA Bridge and Lightweight HPS-to-FPGA Bridge • "Controlling Quality of Service from Software": Added note about register access • "Configuring SDRAM Burst Sizes": Refers to guidelines for selecting burst sizes • "Sharing I/O between the EMIF and the FPGA": New section, discusses constraints on sharing I/Os with EMIF
May 2016	2016.05.27	Maintenance release
May 2016	2016.05.03	Maintenance release
November 2015	2015.11.02	• Correct size of HPS-to-FPGA region in "MPU Address Space" • Clarify power domains in "Functional Description of the SDRAM L3 Interconnect"
May 2015	2015.05.04	• Added address maps and register definitions • Added information about the SDRAM scheduler • Added information about rate adapters • Added information about the observation network
December 2014	2014.12.15	• Added register details for address remapping • Added information about quality of service • Added information about arbitration • Clarified block diagram of SDRAM L3 interconnect
August 2014	2014.08.18	Initial release

HPS-FPGA Bridges 8

2016.10.28

a10_5v4 Subscribe Send Feedback

This chapter describes the bridges in the hard processor system (HPS) used to communicate data between the FPGA fabric and HPS logic.

These bridges make use of Advanced Microcontroller Bus Architecture (AMBA®) Advanced eXtensible Interface (AXI™) protocol, based on Arteris FlexNoC™ network-on-chip (NOC) interconnect IP.

The HPS contains the following HPS-FPGA bridges:

- FPGA-to-HPS Bridge
- HPS-to-FPGA Bridge
- Lightweight HPS-to-FPGA Bridge

Related Information

- **www.arteris.com**
  For detailed information about the FlexNoC Network-on-Chip Interconnect, refer to the Arteris website.
- **SoC Security** on page 6-1
  Information about security features in the HPS-FPGA bridges
- **http://infocenter.arm.com**
  Additional information is available in the AMBA AXI Protocol Specification v1.0, which you can download from the ARM Infocenter website.

Features of the HPS-FPGA Bridges

The HPS-FPGA bridges allow masters in the FPGA fabric to communicate with slaves in the HPS logic and vice versa. For example, if you implement memories or peripherals in the FPGA fabric, HPS components such as the MPU can access them. Components implemented in the FPGA fabric, such as the Nios® II processor, can also access memories and peripherals in the HPS logic.

Table 8-1: HPS-FPGA Bridge Features

Feature	FPGA-to-HPS Bridge	HPS-to-FPGA Bridge	Lightweight HPS-to-FPGA Bridge
Supports the AMBA AXI3 interface and FlexNoC low latency interface protocols	Y	Y	Y
Implements clock crossing and manages the transfer of data across the clock domains in the HPS logic and the FPGA fabric	Y	Y	Y
Performs data width conversion between the HPS logic and the FPGA fabric	Y	Y	Y
Allows configuration of FPGA interface widths at instantiation time	Y	Y	N

Each bridge consists of a master-slave pair with one interface exposed to the FPGA fabric and the other exposed to the HPS logic. The FPGA-to-HPS bridge exposes an AXI slave interface that you can connect to AXI or Avalon-MM master interfaces in the FPGA fabric. The HPS-to-FPGA and lightweight HPS-to-FPGA bridges expose an AXI master interface that you can connect to AXI or Avalon-MM slave interfaces in the FPGA fabric.

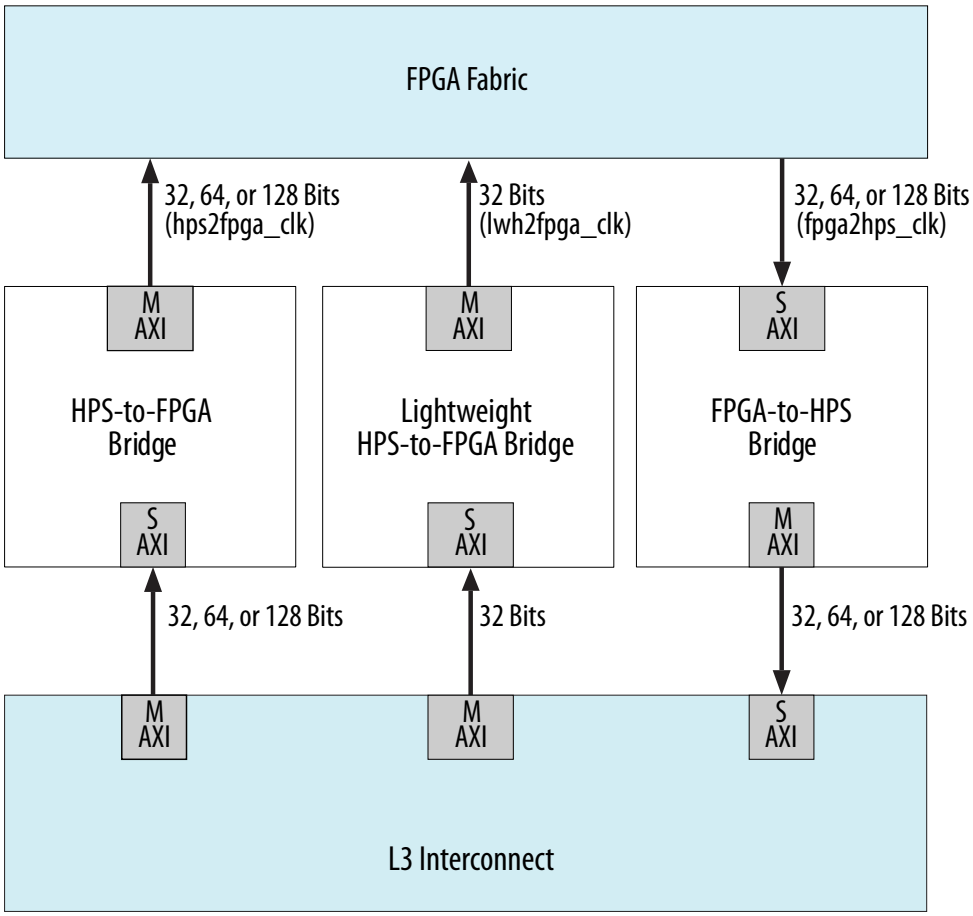
Related Information

AXI Bridges on page 27-10
Information about configuring the AXI bridges

Arria 10 HPS-FPGA Bridges Block Diagram and System Integration

Figure 8-1: HPS-FPGA Bridge Connectivity

The following figure shows the HPS-FPGA bridges in the context of the FPGA fabric and the L3 interconnect to the HPS. Each master (M) and slave (S) interface is shown with its data width(s). The clock domain for each interconnect is shown in parentheses.



The HPS-to-FPGA and lightweight HPS-to-FPGA bridges are both mastered by the level 3 (L3) interconnect. The FPGA-to-HPS bridge masters the L3 interconnect. This arrangement allows any master implemented in the FPGA fabric to access most slaves in the HPS. For example, the FPGA-to-HPS bridge can access the accelerator coherency port (ACP) of the MPU subsystem to perform cache-coherent accesses to the SDRAM subsystem.

Functional Description of the HPS-FPGA Bridges

Functional Description of the FPGA-to-HPS Bridge

The FPGA-to-HPS bridge provides access to the peripherals in the HPS from the FPGA. This access is available to any master implemented in the FPGA fabric. You can configure the bridge slave, which is

exposed to the FPGA fabric, to support the AXI protocol, with a data width of 32, 64 or 128 bits. The FPGA-to-HPS bridge multiplexes the configured data width from the L3 interconnect to the FPGA interface.

Table 8-2: FPGA-to-HPS Bridge Properties

The following table lists the properties of the FPGA-to-HPS bridge, including the configurable slave interface exposed to the FPGA fabric.

Bridge Property	Value
Data width[17]	32, 64, or 128 bits
Clock domain	`fpga2hps_clk`
Byte address width	32 bits
ID width	4 bits
Read acceptance	8 transactions
Write acceptance	8 transactions
Total acceptance	16 transactions

The FPGA-to-HPS bridge is configurable in the HPS component parameter editor, available in Qsys and the IP Catalog. The HPS component parameter editor allows you to set the data path width and the bridge protocol, according to the FPGA bitstream.

Warning: To avoid memory aliasing issues, ensure that Avalon-MM burst transactions into the HPS do not cross the 4 KB address boundary restriction specified by the AXI protocol.

FPGA-to-HPS Access to ACP

When the error correction code (ECC) option is enabled in the level 2 (L2) cache controller, all accesses from the FPGA-to-HPS bridge to the ACP must be 64 bits wide and aligned on 8-byte boundaries after up- or downsizing takes place. Ensure that the address alignment is a multiple of 8 bytes and that (burst size) × (burst length) is a multiple of 8 bytes.

FPGA-to-HPS Bridge Slave Signals

The FPGA-to-HPS bridge slave address channels support user-sideband signals, routed to the ACP in the MPU subsystem. All the signals have a fixed width except the data and write strobes for the read and write data channels. The variable width signals depend on the data width setting of the bridge.

The FPGA-to-HPS bridge incorporates ARM's TrustZone technology by providing the `ARPROT[1]` and `AWPROT[1]` signals, which specify whether a transaction is secure or nonsecure. The firewall logic uses the `AxPROT` signals to determine if a bus transaction matches the security level allowed. You can program security values in the Secure Configuration Registers of the system interconnect.

All peripheral slaves and memories in the SoC are secure when they are released from reset.

[17] The bridge slave data width is user-configurable at the time you instantiate the HPS component in your system.

The following tables list all the signals exposed by the FPGA-to-HPS slave interface to the FPGA fabric.

Table 8-3: FPGA-to-HPS Bridge Slave Write Address Channel Signals

Signal	Width	Direction	Description
AWID	8 bits	Input	Write address ID
AWADDR	32 bits	Input	Write address
AWLEN	4 bits	Input	Burst length
AWSIZE	3 bits	Input	Burst size
AWBURST	2 bits	Input	Burst type
AWLOCK	2 bits	Input	Lock type—Valid values are 00 (normal access) and 01 (exclusive access)
AWCACHE	4 bits	Input	Cache policy type
AWPROT	3 bits	Input	Protection type
AWVALID	1 bit	Input	Write address channel valid
AWREADY	1 bit	Output	Write address channel ready
AWUSER	5 bits	Input	User sideband signals

Table 8-4: FPGA-to-HPS Bridge Slave Write Data Channel Signals

Signal	Width	Direction	Description
WID	8 bits	Input	Write ID
WDATA	32, 64, or 128 bits	Input	Write data
WSTRB	4, 8, or 16 bits	Input	Write data strobes
WLAST	1 bit	Input	Write last data identifier
WVALID	1 bit	Input	Write data channel valid
WREADY	1 bit	Output	Write data channel ready

Table 8-5: FPGA-to-HPS Bridge Slave Write Response Channel Signals

Signal	Width	Direction	Description
BID	8 bits	Output	Write response ID
BRESP	2 bits	Output	Write response
BVALID	1 bit	Output	Write response channel valid
BREADY	1 bit	Input	Write response channel ready

Table 8-6: FPGA-to-HPS Bridge Slave Read Address Channel Signals

Signal	Width	Direction	Description
ARID	8 bits	Input	Read address ID
ARADDR	32 bits	Input	Read address
ARLEN	4 bits	Input	Burst length
ARSIZE	3 bits	Input	Burst size
ARBURST	2 bits	Input	Burst type
ARLOCK	2 bits	Input	Lock type—Valid values are 00 (normal access) and 01 (exclusive access)
ARCACHE	4 bits	Input	Cache policy type
ARPROT	3 bits	Input	Protection type
ARVALID	1 bit	Input	Read address channel valid
ARREADY	1 bit	Output	Read address channel ready
ARUSER	5 bits	Input	Read user sideband signals

Table 8-7: FPGA-to-HPS Bridge Slave Read Data Channel Signals

Signal	Width	Direction	Description
RID	8 bits	Output	Read ID
RDATA	32, 64, or 128 bits	Output	Read data
RRESP	2 bits	Output	Read response
RLAST	1 bit	Output	Read last data identifier

Signal	Width	Direction	Description
RVALID	1 bit	Output	Read data channel valid
RREADY	1 bit	Input	Read data channel ready

Related Information

SoC Security on page 6-1
More information about Secure Firewall Bus Transactions

Functional Description of the HPS-to-FPGA Bridge

The HPS-to-FPGA bridge provides a configurable-width, high-performance master interface to the FPGA fabric. The bridge provides most masters in the HPS with access to logic and peripherals implemented in the FPGA. The effective size of the address space is 0x3FFF0000, or 1 gigabyte (GB) minus the 64 megabytes (MB) occupied by peripherals, lightweight HPS-to-FPGA bridge, and on-chip RAM in the HPS. You can configure the bridge master exposed to the FPGA fabric for 32-, 64-, or 128-bit data. The amount of address space exposed to the MPU subsystem can also be reduced through the L2 cache address filtering mechanism.

The HPS-to-FPGA bridge multiplexes the configured data width from the L3 interconnect to the FPGA interface. The bridge provides width adaptation and clock crossing logic that allows the logic in the FPGA to operate in any clock domain, asynchronous from the HPS.

Warning: The HPS-to-FPGA bridge is accessed if the MPU boots from the FPGA. Before the MPU boots from the FPGA, the FPGA portion of the SoC device must be configured, and the HPS-to-FPGA bridge must be remapped into addressable space. Otherwise, access to the HPS-to-FPGA bridge during the boot process results in a bus error.

Table 8-8: HPS-to-FPGA Bridge Properties

The following table lists the properties of the HPS-to-FPGA bridge, including the configurable master interface exposed to the FPGA fabric.

Bridge Property	Value
Data width [18]	32, 64, or 128 bits
Clock domain	hps2fpga_clk
Byte address width	30 bits
ID width	4 bits
Read acceptance	16 transactions
Write acceptance	16 transactions
Total acceptance	16 transactions

[18] The bridge master data width is user-configurable at the time you instantiate the HPS component in your system.

The HPS-to-FPGA bridge is configurable in the HPS component parameter editor, available in Qsys and the IP Catalog. The HPS component parameter editor allows you to set the data path width and the bridge protocol, according to the FPGA bitstream.

Related Information

- **Functional Description of the System Interconnect** on page 7-12
  Detailed information about connectivity, such as which masters have access to each bridge
- **Cortex-A9 Microprocessor Unit Subsystem** on page 9-1
  Details about L2 cache address filtering
- **AXI Bridges** on page 27-10
  Information about configuring the AXI bridges

HPS-to-FPGA Bridge Master Signals

All the HPS-to-FPGA bridge master signals have a fixed width except the data and write strobes for the read and write data channels. The variable-width signals depend on the data width setting of the bridge interface exposed to the FPGA logic.

The HPS-to-FPGA bridge incorporates ARM's TrustZone technology by providing the `ARPROT[1]` and `AWPROT[1]` signals, which specify whether a transaction is secure or nonsecure. The firewall logic uses the `AxPROT` signals to determine if a bus transaction matches the security level allowed. You can program security values in the Secure Configuration Registers of the system interconnect.

All peripheral slaves and memories in the SoC are secure when they are released from reset.

The following tables list all the signals exposed by the HPS-to-FPGA master interface to the FPGA fabric.

Table 8-9: HPS-to-FPGA Bridge Master Write Address Channel Signals

Signal	Width	Direction	Description
`AWID`	12 bits	Output	Write address ID
`AWADDR`	30 bits	Output	Write address
`AWLEN`	4 bits	Output	Burst length
`AWSIZE`	3 bits	Output	Burst size
`AWBURST`	2 bits	Output	Burst type
`AWLOCK`	2 bits	Output	Lock type—Valid values are 00 (normal access) and 01 (exclusive access)
`AWCACHE`	4 bits	Output	Cache policy type
`AWPROT`	3 bits	Output	Protection type
`AWVALID`	1 bit	Output	Write address channel valid
`AWREADY`	1 bit	Input	Write address channel ready

Table 8-10: HPS-to-FPGA Bridge Master Write Data Channel Signals

Signal	Width	Direction	Description
WID	12 bits	Output	Write ID
WDATA	32, 64, or 128 bits	Output	Write data
WSTRB	4, 8, or 16 bits	Output	Write data strobes
WLAST	1 bit	Output	Write last data identifier
WVALID	1 bit	Output	Write data channel valid
WREADY	1 bit	Input	Write data channel ready

Table 8-11: HPS-to-FPGA Bridge Master Write Response Channel Signals

Signal	Width	Direction	Description
BID	12 bits	Input	Write response ID
BRESP	2 bits	Input	Write response
BVALID	1 bit	Input	Write response channel valid
BREADY	1 bit	Output	Write response channel ready

Table 8-12: HPS-to-FPGA Bridge Master Read Address Channel Signals

Signal	Width	Direction	Description
ARID	12 bits	Output	Read address ID
ARADDR	30 bits	Output	Read address
ARLEN	4 bits	Output	Burst length
ARSIZE	3 bits	Output	Burst size
ARBURST	2 bits	Output	Burst type
ARLOCK	2 bits	Output	Lock type—Valid values are 00 (normal access) and 01 (exclusive access)
ARCACHE	4 bits	Output	Cache policy type
ARPROT	3 bits	Output	Protection type

Signal	Width	Direction	Description
ARVALID	1 bit	Output	Read address channel valid
ARREADY	1 bit	Input	Read address channel ready

Table 8-13: HPS-to-FPGA Bridge Master Read Data Channel Signals

Signal	Width	Direction	Description
RID	12 bits	Input	Read ID
RDATA	32, 64, or 128 bits	Input	Read data
RRESP	2 bits	Input	Read response
RLAST	1 bit	Input	Read last data identifier
RVALID	1 bit	Input	Read data channel valid
RREADY	1 bit	Output	Read data channel ready

Related Information

SoC Security on page 6-1
More information about Secure Firewall Bus Transactions

Functional Description of the Lightweight HPS-to-FPGA Bridge

The lightweight HPS-to-FPGA bridge provides a lower-performance interface to the FPGA fabric. This interface is useful for accessing the control and status registers of soft peripherals. The bridge provides a 2 MB address space and access to logic, peripherals, and memory implemented in the FPGA fabric. The MPU subsystem, direct memory access (DMA) controller, and debug access port (DAP) can use the lightweight HPS-to-FPGA bridge to access the FPGA fabric or NoC registers.

The lightweight HPS-to-FPGA bridge has a fixed data width of 32 bits.

Use the lightweight HPS-to-FPGA bridge as a secondary, lower-performance master interface to the FPGA fabric. With a fixed width and a smaller address space, the lightweight bridge is useful for low-bandwidth traffic, such as memory-mapped register accesses to FPGA peripherals. This approach diverts traffic from the high-performance HPS-to-FPGA bridge, and can improve both register access latency and overall system performance.

Warning: Before the lightweight HPS-to-FPGA bridge can be accessed, the FPGA portion of the SoC device must be configured, and the lightweight HPS-to-FPGA bridge must be remapped into addressable space. Otherwise, accesses to the lightweight HPS-to-FPGA bridge will result in a bus error.

Table 8-14: Lightweight HPS-to-FPGA Bridge Properties

This table lists the properties of the lightweight HPS-to-FPGA bridge, including the master interface exposed to the FPGA fabric.

Bridge Property	Value
Data width	32 bits
Clock domain	lwh2fpga_clk
Byte address width	21 bits
ID width	4 bits
Read acceptance	16 transactions
Write acceptance	16 transactions
Total acceptance	16 transactions

The lightweight HPS-to-FPGA bridge is configurable in the HPS component parameter editor, available in Qsys and the IP Catalog. The HPS component parameter editor allows you to set the bridge protocol, according to the FPGA bitstream.

Related Information

- **Arria 10 HPS-FPGA Bridges Block Diagram and System Integration** on page 8-3
  Figure showing the lightweight HPS-to-FPGA bridge's three master interfaces
- **Functional Description of the System Interconnect** on page 7-12
  Detailed information about connectivity, such as which masters have access to each bridge
- **AXI Bridges** on page 27-10
  Information about configuring the AXI bridges

Lightweight HPS-to-FPGA Bridge Master Signals

All the lightweight HPS-to-FPGA bridge master signals have a fixed width.

The lightweight HPS-to-FPGA bridge incorporates ARM's TrustZone technology by providing the ARPROT[1] and AWPROT[1] signals, which specify whether a transaction is secure or nonsecure. The firewall logic uses the AxPROT signals to determine if a bus transaction matches the security level allowed. You can program security values in the Secure Configuration Registers of the system interconnect.

All peripheral slaves and memories in the SoC are secure when they are released from reset.

The following tables list all the signals exposed by the lightweight HPS-to-FPGA master interface to the FPGA fabric.

Table 8-15: Lightweight HPS-to-FPGA Bridge Master Write Address Channel Signals

Signal	Width	Direction	Description
AWID	12 bits	Output	Write address ID
AWADDR	21 bits	Output	Write address
AWLEN	4 bits	Output	Burst length

Signal	Width	Direction	Description
AWSIZE	3 bits	Output	Burst size
AWBURST	2 bits	Output	Burst type
AWLOCK	2 bits	Output	Lock type—Valid values are 00 (normal access) and 01 (exclusive access)
AWCACHE	4 bits	Output	Cache policy type
AWPROT	3 bits	Output	Protection type
AWVALID	1 bit	Output	Write address channel valid
AWREADY	1 bit	Input	Write address channel ready

Table 8-16: Lightweight HPS-to-FPGA Bridge Master Write Data Channel Signals

Signal	Width	Direction	Description
WID	12 bits	Output	Write ID
WDATA	32 bits	Output	Write data
WSTRB	4 bits	Output	Write data strobes
WLAST	1 bit	Output	Write last data identifier
WVALID	1 bit	Output	Write data channel valid
WREADY	1 bit	Input	Write data channel ready

Table 8-17: Lightweight HPS-to-FPGA Bridge Master Write Response Channel Signals

Signal	Width	Direction	Description
BID	12 bits	Input	Write response ID
BRESP	2 bits	Input	Write response
BVALID	1 bit	Input	Write response channel valid
BREADY	1 bit	Output	Write response channel ready

Table 8-18: Lightweight HPS-to-FPGA Bridge Master Read Address Channel Signals

Signal	Width	Direction	Description
ARID	12 bits	Output	Read address ID
ARADDR	21 bits	Output	Read address
ARLEN	4 bits	Output	Burst length
ARSIZE	3 bits	Output	Burst size
ARBURST	2 bits	Output	Burst type
ARLOCK	2 bits	Output	Lock type—Valid values are 00 (normal access) and 01 (exclusive access)
ARCACHE	4 bits	Output	Cache policy type
ARPROT	3 bits	Output	Protection type
ARVALID	1 bit	Output	Read address channel valid
ARREADY	1 bit	Input	Read address channel ready

Table 8-19: Lightweight HPS-to-FPGA Bridge Master Read Data Channel Signals

Signal	Width	Direction	Description
RID	12 bits	Input	Read ID
RDATA	32 bits	Input	Read data
RRESP	2 bits	Input	Read response
RLAST	1 bit	Input	Read last data identifier
RVALID	1 bit	Input	Read data channel valid
RREADY	1 bit	Output	Read data channel ready

Related Information

SoC Security on page 6-1
More information about Secure Firewall Bus Transactions

Clocks and Resets

FPGA-to-HPS Bridge Clocks and Resets

The master interface of the bridge in the HPS logic operates in the `l3_main_clk` clock domain. The slave interface exposed to the FPGA fabric operates in the `fpga2hps_clk` clock domain provided by the user logic. The bridge provides clock crossing logic that allows the logic in the FPGA to operate in any clock domain, asynchronous from the HPS.

The FPGA-to-HPS bridge has one reset signal, `fpga2hps_bridge_rst_n`. The reset manager drives this signal to the FPGA-to-HPS bridge on a cold or warm reset.

Related Information

- **Clock Manager** on page 2-1
- **HPS Component Interfaces** on page 28-1
  Information about the HPS-FPGA bridge clock interfaces

HPS-to-FPGA Bridge Clocks and Resets

The master interface into the FPGA fabric operates in the `hps2fpga_clk` clock domain. The clock is provided by user logic. The slave interface of the bridge in the HPS logic operates in the `l3_main_clk` clock domain. The bridge provides clock crossing logic that allows the logic in the FPGA to operate in any clock domain, asynchronous from the HPS.

The HPS-to-FPGA bridge has one reset signal, `hps2fpga_bridge_rst_n`. The reset manager drives this signal to the HPS-to-FPGA bridge on a cold or warm reset.

Related Information

- **Clock Manager** on page 2-1
- **HPS Component Interfaces** on page 28-1
  Information about the HPS-FPGA bridge clock interfaces

Lightweight HPS-to-FPGA Bridge Clocks and Resets

The master interface into the FPGA fabric operates in the `lwh2fpga_clk` clock domain. The clock is provided by custom logic in the FPGA fabric. The slave interface of the bridge in the HPS logic operates in the `l3_main_clk` clock domain. The bridge provides clock crossing logic that allows the logic in the FPGA to operate in any clock domain, asynchronous from the HPS.

The lightweight HPS-to-FPGA bridge has one reset signal, `lwhps2fpga_bridge_rst_n`. The reset manager drives this signal to the lightweight HPS-to-FPGA bridge on a cold or warm reset.

Related Information

- **Clock Manager** on page 2-1
- **HPS Component Interfaces** on page 28-1
  Information about the HPS-FPGA bridge clock interfaces

Taking HPS-FPGA Bridges Out of Reset

When a cold or warm reset is issued in the HPS, the reset manager resets this module and holds it in reset until software releases it.

After the Cortex-A9 MPCore CPU boots, it can deassert the reset signal by clearing the appropriate bits in the reset manager's corresponding reset register. For details about reset registers, refer to "Module Reset Signals".

Related Information

Modules Requiring Software Deassert on page 3-13
Reset register names

Data Width Sizing

The HPS-to-FPGA and FPGA-to-HPS bridges allow 32-, 64-, and 128-bit interfaces to be exposed to the FPGA fabric. For 32-bit and 128-bit interfaces, the bridge performs data width conversion to the fixed 64-bit interface within the HPS. This conversion is called *upsizing* in the case of data being converted from a 64-bit interface to a 128-bit interface. It is called *downsizing* in the case of data being converted from a 64-bit interface to a 32-bit interface. If an exclusive access is split into multiple transactions, the transactions lose their exclusive access information.

During the upsizing or downsizing process, transactions can also be resized using a data merging technique. For example, in the case of a 32-bit to 64-bit upsizing, if the size of each beat entering the bridge's 32-bit interface is only two bytes, the bridge can merge up to four beats to form a single 64-bit beat. Similarly, in the case of a 128-bit to 64-bit downsizing, if the size of each beat entering the bridge's 128-bit interface is only four bytes, the bridge can merge two beats to form a single 64-bit beat.

Ready Latency Support

The HPS-to-FPGA and FPGA-to-HPS bridges support an optional ready latency feature, which allows your design to run at a higher F_{MAX}. When enabled, this feature adds a pipeline stage between the HPS and the FPGA fabric to improve ready/valid handshake timing. Enabling this feature entails a minimal increase in bridge latency.

You can enable this feature when you instantiate the HPS component in Qsys.

Related Information

AXI Bridges on page 27-10
Information about configuring the AXI bridges

HPS-FPGA Bridges Address Map and Register Definitions for Arria 10

For complete HPS address map and register definitions, refer to the **Arria 10 HPS Address Map and Register Definitions**.

fpga_bridge_soc2fpga128 Address Map

This address space is allocated for the configurable-width, high-performance master interface to the FPGA fabric. For more information about the HPS-to-FPGA bridges, refer to the HPS-FPGA Bridges chapter in the Arria 10 Hard Processor System Technical Reference Manual.

Module Instance	Base Address	End Address
i_fpga_bridge_soc2fpga128	0xC0000000	0xFBFFFFFF

fpga_bridge_lwsoc2fpga Address Map

This address space is allocated for FPGA-configured slaves driven by the lightweight HPS-to-FPGA bridge master. Address assignment within this space is user defined. For more information about Lightweight HPS-to-FPGA bridges, refer to the HPS-FPGA Bridges chapter of the Arria 10 Hard Processor System Technical Reference Manual.

Module Instance	Base Address	End Address
`i_fpga_bridge_lwsoc2fpga`	`0xFF200000`	`0xFF3FFFFF`

Document Revision History

Table 8-20: Document Revision History

Date	Version	Changes
October 2016	2016.10.28	• Added note about AXI 4 KB boundary restriction • Clarified description of bridge master-slave connections
May 2016	2016.05.27	Maintenance release
November 2015	2015.11.02	Maintenance release
May 2015	2015.05.04	Added address maps and register definitions
December 2014	2014.12.15	Maintenance release
August 2014	2014.08.18	Initial release.

Cortex-A9 Microprocessor Unit Subsystem

9

2016.10.28

a10_5v4

Subscribe Send Feedback

The hard processor system (HPS) in the Altera SoC device includes a stand-alone, full-featured ARM Cortex-A9, dual-core 32-bit application processor. The Cortex-A9 microprocessor unit (MPU) subsystem is composed of a Cortex-A9 MPCore, a level 2 (L2) cache, and debugging modules.

Features of the Cortex-A9 MPU Subsystem

The Altera Cortex-A9 MPU subsystem provides the following features:

- Two ARM Cortex-A9 processors, each with the following support modules:
  - ARM NEON single instruction, multiple data (SIMD) coprocesoor
  - Memory Management Unit (MMU)
  - 32 KB instruction cache
  - 32 KB data cache
  - Private interval timer
  - Private watchdog timer
- Interrupt controller
- Global timer
- Snoop control unit (SCU)
- Accelerator Coherency Port (ACP)
- ARM L2 cache controller
- TrustZone system security extensions
- Symmetric multiprocessing (SMP) and asymmetric multiprocessing (AMP) modes
- Debugging modules

The Cortex-A9 MPU incorporates the following core and L2 cache versions:

Table 9-1: Cortex-A9 MPU Module Versions

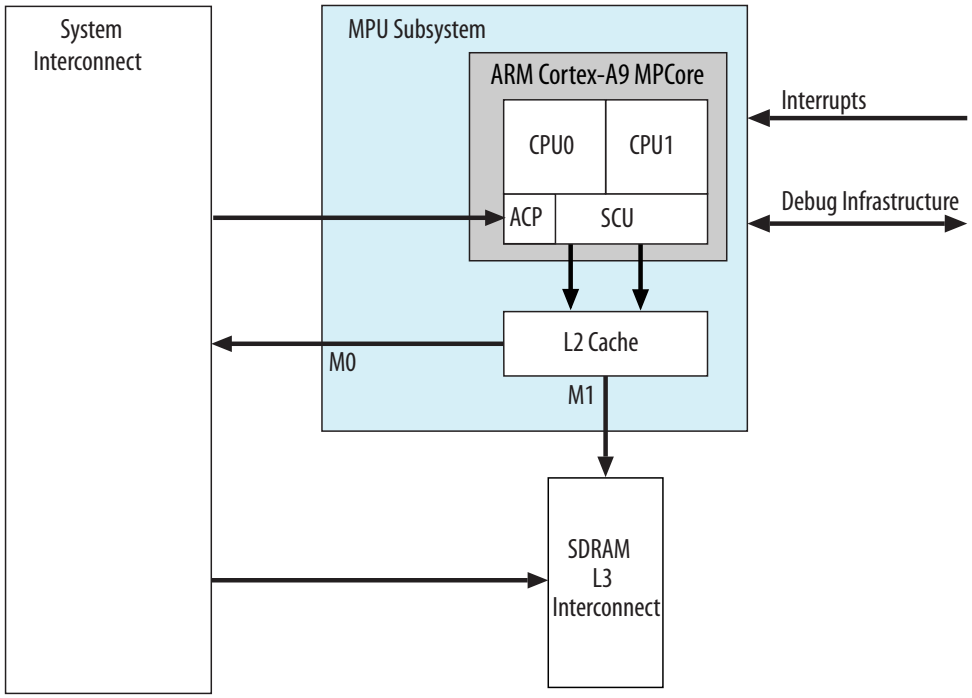
Feature	Version
Cortex-A9 Core	r4p1
L2-310 Level 2 Cache Controller	r3p3

Cortex-A9 MPU Subsystem Block Diagram and System Integration

Cortex-A9 MPU Subsystem with System Interconnect

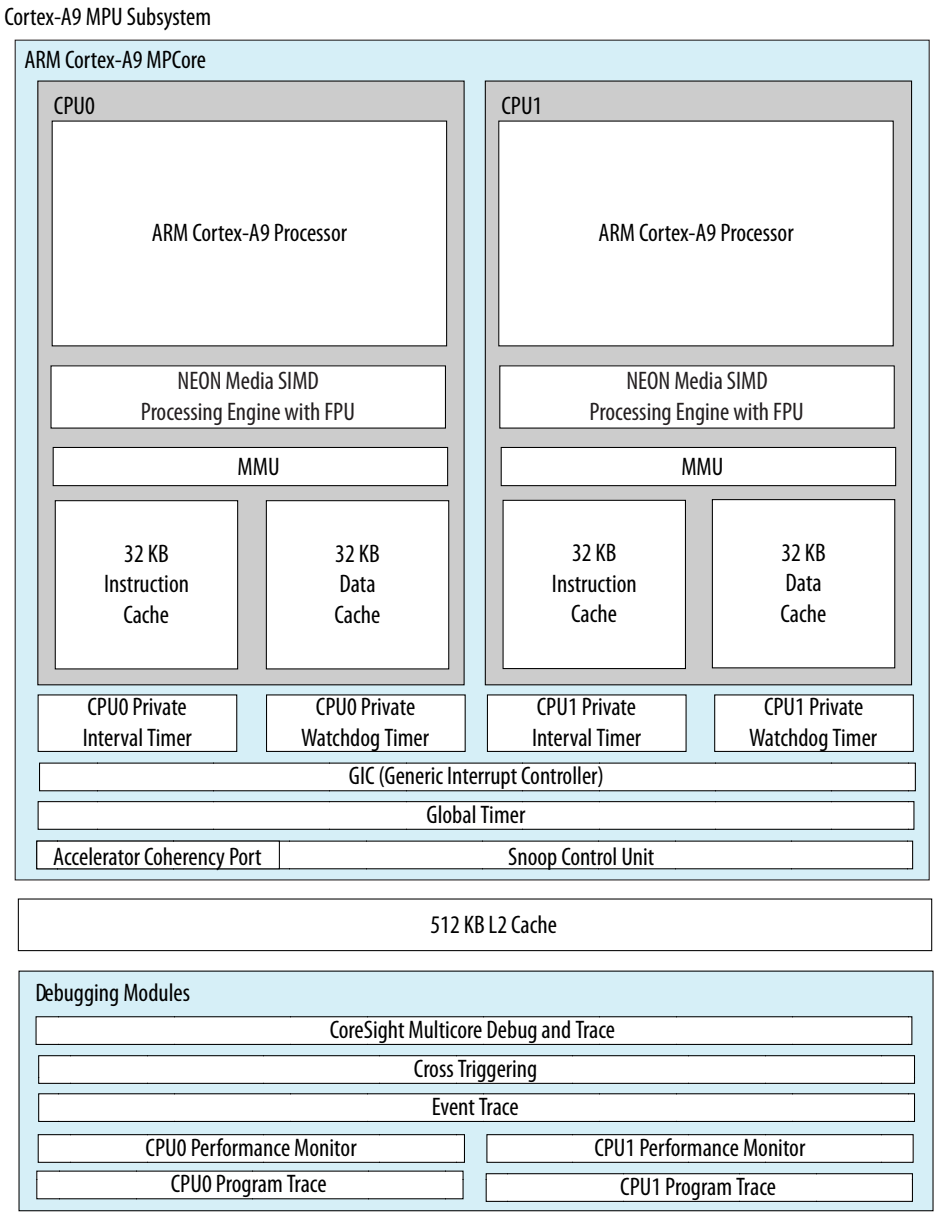
This block diagram shows a dual-core MPU subsystem in the context of the HPS, with the L2 cache. The L2 cache can access either the system interconnect or the SDRAM L3 Interconnect.

Figure 9-1: Cortex-A9 MPU Subsystem with Interconnect Block Diagram



Cortex-A9 MPU Subsystem Internals

Figure 9-2: Cortex-A9 MPU Subsystem Block Diagram



Cortex-A9 MPCore

The MPU subsystem includes a stand-alone, full-featured ARM Cortex-A9 MPCore dual-core 32-bit application processor. The processor, like other HPS masters, can access soft IP in the FPGA fabric through the HPS-to-FPGA bridges.

Functional Description

The ARM Cortex-A9 MPCore contains the following sub-modules:

- Two Cortex-A9 Revision r4p1 processors operating in SMP or AMP mode
- Snoop control unit (SCU)
- Private interval timer for each processor core
- Private watchdog timer for each processor core
- Global timer
- Interrupt controller

Each transaction originating from the Altera Cortex-A9 MPU subsystem can be flagged as secure or non-secure.

Related Information

Cortex-A9 MPU Subsystem Register Implementation on page 9-46
For more information regarding the Cortex MPU Subsystem Address Map, refer to the *Cortex-A9 MPU Subsystem Register Implementation* section.

Implementation Details

Table 9-2: Cortex-A9 MPCore Processor Configuration

This table shows the parameter settings for the Altera Cortex-A9 MPCore.

Feature	Options
Cortex-A9 processors	2
Instruction cache size per Cortex-A9 processor	32 KB
Data cache size per Cortex-A9 processor	32 KB
TLB size per Cortex-A9 processor	512 entries
Media Processing Engine with NEON™ technology per Cortex-A9 processor(19)	Included
Preload Engine per Cortex-A9 processor	Included
Number of entries in the Preload Engine FIFO per Cortex-A9 processor	16
Jazelle DBX extension per Cortex-A9 processor	Full
Program Trace Macrocell (PTM) interface per Cortex-A9 processor	Included
Support for parity error detection (20)	Included
Master ports	Two
Accelerator Coherency Port	Included

Related Information

ARM Infocenter

For more information regarding the Cortex-A9 MPCore features and functions.

(19) Includes support for floating-point operations.

(20) For a description of the parity error scheme and parity error signals, refer to the Cortex-A9 Technical Reference Manual, available on the ARM website (infocenter.arm.com).

Cortex-A9 Processor

Each Cortex-A9 processor includes the following hardware blocks:

- ARM NEON™ single instruction, multiple data (SIMD) coprocessor with vector floating-point (VFP) v3 double-precision floating point unit for media and signal processing acceleration
  - Single- and double-precision IEEE-754 floating point math support
  - Integer and polynomial math support
- Level 1 (L1) cache with parity checking
  - 32 KB four-way set-associative instruction cache
  - 32 KB four-way set-associative data cache
- CoreSight™ Program Trace Macrocell (PTM) supporting instruction trace

Each Cortex-A9 processor supports the following features:

- Dual-issue superscalar pipeline with advanced branch prediction
- Out-of-order (OoO) dispatch and speculative instruction execution
- 2.5 million instructions per second (MIPS) per MHz, based on the Dhrystone 2.1 benchmark
- 2048-entry Branch Target Address Cache (BTAC)
- 512-entry translation lookaside buffer (TLB)
- 64-entry instruction micro-TLB
- TrustZone security extensions
- Configurable data endianness
- Jazelle® DBX Extensions for byte-code dynamic compiler support
- The Cortex-A9 processor architecture supports the following instruction sets:
  - The ARMv7-A performance-optimized instruction set
  - The memory-optimized Thumb®-2 mixed instruction set
    - Improves energy efficiency
    - 31% smaller memory footprint
    - 38% faster than the original Thumb instruction set
  - The Thumb instruction set—supported for legacy applications
- Each processor core in the Altera HPS includes an MMU to support the memory management requirements of common modern operating systems.

The Cortex-A9 processors are designated CPU0 and CPU1.

Related Information

ARM Infocenter

Detailed documentation of ARM Cortex-A9 series processors is available on the ARM Infocenter website.

Reset

When a cold or warm reset is issued in the MPU, CPU0 is released from reset automatically. If CPU1 is present, then its reset signals are left asserted when a cold or warm reset is issued. After CPU0 comes out of reset, it can deassert CPU1's reset signals by clearing the `CPU1` bit in the MPU Module Reset (`mpumodrst`) register in the Reset Manager.

Related Information

Reset Manager on page 3-1

Interactive Debugging Features

Each Cortex-A9 processor has built-in debugging capabilities, including six hardware breakpoints (two with Context ID comparison capability) and four watchpoints. The interactive debugging features can be controlled by external JTAG tools or by processor-based monitor code.

Related Information

ARM Infocenter

For more information about the interactive debugging system, refer to the *Debug* chapter of the Cortex-A9 Technical Reference Manual, available on the ARM Infocenter website.

L1 Caches

Cache memory that is closely coupled with an associated processor is called level 1, or L1 cache. Each Cortex-A9 processor has two independent 32 KB L1 caches—one for instructions and one for data—allowing simultaneous instruction fetches and data access. Each L1 cache is four-way set associative, with 32 bytes per line, and supports parity checking.

Note: A parity error cannot be recovered and is indicated by one of the parity error interrupt signals. On a parity error interrupt, you can reset the system or perform further actions depending on the indication of the interrupt signals.

Cache Latency

Latency for cache hits and misses varies.

The latency for an L1 cache hit is 1 clock. The latency for an L1 cache miss and L2 cache hit is 6 clocks best case. Latency in the L2 cache can vary depending on other operations in the L2. Parity and ECC settings have no effect on latency. A single-bit ECC error is corrected during the L2 read, but is not re-written to the L2 RAM.

Preload Engine

The preload engine (PLE) is a hardware block that enables the L2 cache to preload selected regions of memory.

The PLE signals the L2 cache when a cache line is needed in the L2 cache, by making the processor data master port start fetching the data. The processor data master does not complete the fetch or return the data to the processor. However, the L2 cache can then proceed to load the cache line. The data is only loaded to the L2 cache, not to the L1 cache or processor registers.

The preload functionality is under software control. The following PLE control parameters must be programmed:

- Programmed parameters, including the following:
  - Base address
  - Length of stride
  - Number of blocks
- A valid bit
- TrustZone memory protection for the cache memory, with an NS (non-secure) state bit
- A translation table base (TTB) address
- An Address Space Identifier (ASID) value

Related Information

ARM Infocenter

For more information about the PLE, refer to the *Preload Engine* chapter of the *Cortex-A9 Technical Reference Manual*, available on the ARM Infocenter website.

Floating Point Unit

Each ARM Cortex-A9 processor includes full support for IEEE-754 floating point operations.

The floating-point unit (FPU) can execute half-, single-, and double-precision variants of the following operations:

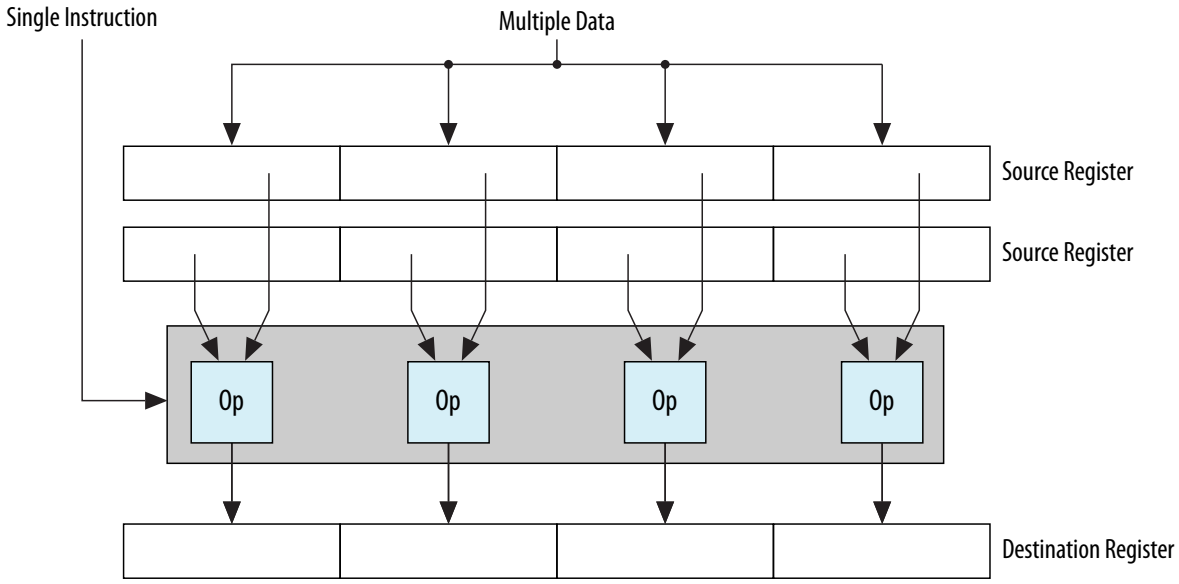
- Add
- Subtract
- Multiply
- Divide
- Multiply and accumulate (MAC)
- Square root

The FPU also converts between floating-point data formats and integers, including special operations to round towards zero required by high-level languages.

NEON Multimedia Processing Engine

The NEON multimedia processing engine (MPE) provides hardware acceleration for media and signal processing applications. Each CPU includes an ARM NEON MPE that supports SIMD processing.

Single Instruction, Multiple Data (SIMD) Processing



Features of the NEON MPE

The NEON processing engine accelerates multimedia and signal processing algorithms such as video encoding and decoding, 2-D and 3-D graphics, audio and speech processing, image processing, telephony, and sound synthesis.

The Cortex-A9 NEON MPE performs the following types of operations:

- SIMD and scalar single-precision floating-point computations
- Scalar double-precision floating-point computation
- SIMD and scalar half-precision floating-point conversion
- 8-, 16-, 32-, and 64-bit signed and unsigned integer SIMD computation
- 8-bit or 16-bit polynomial computation for single-bit coefficients

The following operations are available:

- Addition and subtraction
- Multiplication with optional accumulation (MAC)
- Maximum- or minimum-value driven lane selection operations
- Inverse square root approximation
- Comprehensive data-structure load instructions, including register-bank-resident table lookup

Related Information

ARM Infocenter
For more information about the Cortex-A9 NEON MPE, refer to the Cortex-A9 NEON™ Media Processing Engine Technical Reference Manual, which you can download from the ARM Infocenter website.

Memory Management Unit

The MMU is used in conjunction with the L1 and L2 caches to translate virtual addresses used by software to physical addresses used by hardware. Each processor has a private MMU.

TLBs Supported By the MMU

TLB Type	Memory Type	Number of Entries	Associativity
Micro TLB	Instruction	64	Fully associative
Micro TLB	Data	64	Fully associative
Main TLB	Instruction and Data	512	Two-way associative

TLB Features

The main TLB has the following features:

- Lockable entries using the lock-by-entry model
- Supports hardware page table walks to perform look-ups in the L1 data cache

The MPU address map is divided into the following regions:

- The boot region
- The SDRAM region
- The FPGA slaves region
- The HPS peripherals region

Note: The `SMP` bit in the `ACTLR` register must be set before enabling the MMU.

Related Information

- **ARM Infocenter**
  For more information about the MMU, refer to the *Memory Management Unit* chapter of the *Cortex-A9 Technical Reference Manual*, available on the ARM Infocenter website.
- **Cortex-A9 MPCore Technical Reference Manual**
  For more further information about the `ACTLR register`, refer to the *Cortex-A9 MPCore Technical Reference Manual*, available on the ARM Infocenter website.

The Boot Region

The boot region is 1 MB in size, based at address 0. After power-on, or after reset of the system interconnect, the boot region is occupied by the boot ROM, allowing the Cortex-A9 MPCore to boot. Although the boot region size is 1 MB, accesses beyond 128 KB are illegal because the boot ROM is only 128 KB.

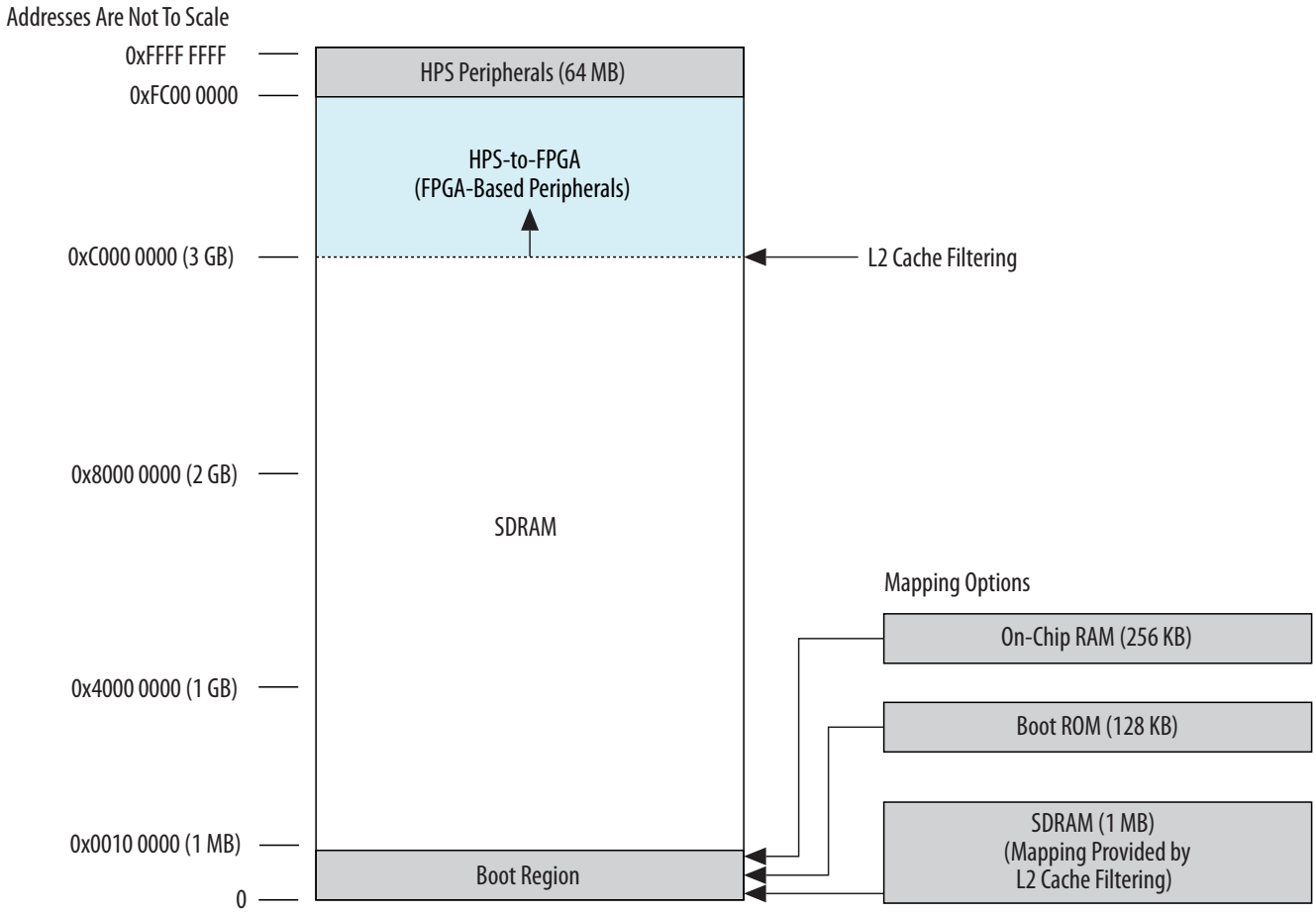
The 1 MB boot region can be subsequently remapped to the bottom 1 MB of SDRAM region by adjusting the L2 cache address filter.

Note: Alternatively, the boot region can be mapped to the 256 KB on-chip RAM.

Related Information

- **Address Remapping** on page 7-19
  For more information about address remapping
- **System Interconnect** on page 7-1

Boot ROM Mapping



The SDRAM Region

The SDRAM region starts at address `0x100000` (1 MB). The top of the region is determined by the L2 cache filter.

The L2 cache contains a filtering mechanism that routes accesses to the SDRAM L3 interconnect and system interconnect. The filter defines a filter range with start and end addresses. Any access within this filter range is routed to the SDRAM L3 interconnect. Accesses outside of this filter range are routed to the system interconnect.

The start and end addresses are specified in the following register fields:

- `reg12_addr_filtering_start.address_filtering_start`
- `reg12_address_filtering_end.address_filtering_end`

To remap the lower 1MB of SDRAM into the boot region, set the filter start address to `0x0` to ensure accesses between `0x0` and `0xFFFFF` are routed to the SDRAM. Independently, you can set the filter end address in 1 MB increments above `0xC0000000` to extend the upper bounds of the SDRAM region. However, you achieve this extended range at the expense of the FPGA peripheral address span. Depending on the address filter settings in the L2 cache, the top of the SDRAM region can range from `0xBFFFFFFF` to `0xFBFFFFFF`.

Related Information

L2 Cache on page 9-34

The FPGA Slaves Region

The Cortex-A9 MPU subsystem supports the variable-sized FPGA slaves region to communicate with FPGA-based peripherals. This region can start as low as `0xC0000000`, depending on the L2 cache filter settings. The top of the FPGA slaves region is located at `0xFBFFFFFF`. As a result, the size of the FPGA slaves region can range from 0 to `0x3C000000` bytes.

The HPS Peripherals Region

The HPS peripherals region is the top 64 MB in the address space, starting at `0xFC000000` and extending to `0xFFFFFFFF`. The HPS peripherals region is always allocated to the HPS dedicated peripherals for the Altera Cortex-A9 MPU subsystem.

Performance Monitoring Unit

Each Cortex-A9 processor has a Performance Monitoring Unit (PMU). The PMU supports 58 events to gather statistics on the operation of the processor and memory system. Six counters in the PMU accumulate the events in real time. The PMU counters are accessible either from the processor itself, using the Coprocessor 14 (CP14) interface, or from an external debugger. The events are also supplied to the PTM and can be used for trigger or trace.

Related Information

ARM Infocenter

For more information about the PMU, refer to the *Performance Monitoring Unit* chapter of the *Cortex-A9 Technical Reference Manual*, available on the ARM Infocenter website.

ARM Cortex-A9 MPCore Timers

There is one interval timer and one watchdog timer for each processor.

Functional Description

Each timer is private, meaning that only its associated processor can access it. If the watchdog timer is not needed, it can be configured as a second interval timer.

Each private interval and watchdog timer has the following features:

- A 32-bit counter that optionally generates an interrupt when it reaches zero
- Configurable starting values for the counter
- An eight-bit prescaler value to qualify the clock period

Implementation Details

The timers are configurable to either single-shot or auto-reload mode. The timer blocks are clocked by `mpu_periph_clk` running at ¼ the rate of the `mpu_clk`.

Related Information

ARM Infocenter
For more information about private timers, refer to "About the private timer and watchdog blocks" in the *Global timer, Private timers, and Watchdog registers* chapter of the *Cortex-A9 MPCore Technical Reference Manual*, available on the ARM Infocenter website.

Generic Interrupt Controller

Functional Description

The PL390 Generic Interrupt Controller (GIC) supports up to 128 interrupt sources, including dedicated peripherals and IP implemented in the FPGA fabric. In a dual-core system, the GIC is shared by both Cortex-A9 processors. Each processor also has 16 banked software-generated interrupts and 16 banked private peripheral interrupts, which occupy GIC interrupt numbers 0 to 31.

Implementation Details

The configuration and control for the GIC is memory-mapped and accessed through the SCU. The GIC are clocked by `mpu_periph_clk`, running at ¼ the rate of `mpu_clk`.

Related Information

- **GIC Interrupt Map for the Arria 10 SoC HPS** on page 9-13
- **ARM Infocenter**
  For more information about the PL390 GIC, refer to the *Interrupt Controller* chapter of the *Cortex-A9 MPCore Technical Reference Manual*, available on the ARM Infocenter website.

GIC Interrupt Map for the Arria 10 SoC HPS

Note: To ensure that you are using the correct GIC interrupt number, your code should refer to the symbolic interrupt name, as shown in the **Interrupt Name** column. Symbolic interrupt names are defined in a header file distributed with the source installation for your operating system.

Table 9-3: GIC Interrupt Map

GIC Interrupt Number	Source Block	Interrupt Name	Combined Interrupts	Triggering
32	System Manager	DERR_Global	This interrupt combines: ddr_derr, dma_derr, emac0_tx_derr, emac0_rx_derr, emac1_tx_derr, emac1_rx_derr, emac2_tx_derr, emac2_rx_derr, usb0_derr, usb1_derr, sdmmc_porta_derr, sdmmc_portb_derr, nandr_derr, nandw_derr, nande_derr, qspi_derr, ram_derr, and l2_derr.	Level

GIC Interrupt Number	Source Block	Interrupt Name	Combined Interrupts	Triggering
33	System Manager	cpux_parityfail	This interrupt combines: cpu0_parityfail_BTAC, cpu0_parityfail_GHB, cpu0_parityfail_I_Tag, cpu0_parityfail_I_Data, cpu0_parityfail_TLB, cpu0_parityfail_D_Outer, cpu0_parityfail_D_Tag, cpu0_parityfail_D_Data, cpu1_parityfail_BTAC, cpu1_parityfail_GHB, cpu1_parityfail_I_Tag, cpu1_parityfail_I_Data, cpu1_parityfail_TLB, cpu1_parityfail_D_Outer, cpu1_parityfail_D_Tag, cpu1_parityfail_D_Data, scu_parityfail0, and scu_parityfail1.	Level

GIC Interrupt Number	Source Block	Interrupt Name	Combined Interrupts	Triggering
34	System Manager	SERR_Global	This interrupt combines: ddr_serr, dma_serr, emac0_tx_serr, emac0_rx_serr, emac1_tx_serr, emac1_rx_serr, emac2_tx_serr, emac2_rx_serr, usb0_serr, usb1_serr, sdmmc_porta_serr, sdmmc_portb_serr, nandr_serr, nandw_serr, nande_serr, qspi_serr, ram_serr, and l2_serr.	Level
35	CortexA9_0	cpu0_deflags0	-	Level
36	CortexA9_0	cpu0_deflags1	-	Level
37	CortexA9_0	cpu0_deflags2	-	Level
38	CortexA9_0	cpu0_deflags3	-	Level
39	CortexA9_0	cpu0_deflags4	-	Level
40	CortexA9_0	cpu0_deflags5	-	Level
41	CortexA9_0	cpu0_deflags6	-	Level
42	CortexA9_1	cpu1_deflags0	-	Level

GIC Interrupt Number	Source Block	Interrupt Name	Combined Interrupts	Triggering
43	CortexA9_1	cpu1_deflags1	-	Level
44	CortexA9_1	cpu1_deflags2	-	Level
45	CortexA9_1	cpu1_deflags3	-	Level
46	CortexA9_1	cpu1_deflags4	-	Level
47	CortexA9_1	cpu1_deflags5	-	Level
48	CortexA9_1	cpu1_deflags6	-	Level
49	SCU	scu_ev_abort	-	Edge-triggered
50	L2-Cache	l2_combined_IRQ	This interrupt combines: DECERRINTR, ECNTRINTR, ERRRDINTR, ERRRTINTR, ERRWDINTR, ERRWTINTR, PARRDINTR, PARRTINTR, and SLVERRINTR.	Level
51	FPGA	F2S_FPGA_IRQ0	-	Level or Edge
52	FPGA	F2S_FPGA_IRQ1	-	Level or Edge
53	FPGA	F2S_FPGA_IRQ2	-	Level or Edge
54	FPGA	F2S_FPGA_IRQ3	-	Level or Edge
55	FPGA	F2S_FPGA_IRQ4	-	Level or Edge

GIC Interrupt Number	Source Block	Interrupt Name	Combined Interrupts	Triggering
56	FPGA	F2S_FPGA_IRQ5	-	Level or Edge
57	FPGA	F2S_FPGA_IRQ6	-	Level or Edge
58	FPGA	F2S_FPGA_IRQ7	-	Level or Edge
59	FPGA	F2S_FPGA_IRQ8	-	Level or Edge
60	FPGA	F2S_FPGA_IRQ9	-	Level or Edge
61	FPGA	F2S_FPGA_IRQ10	-	Level or Edge
62	FPGA	F2S_FPGA_IRQ11	-	Level or Edge
63	FPGA	F2S_FPGA_IRQ12	-	Level or Edge
64	FPGA	F2S_FPGA_IRQ13	-	Level or Edge
65	FPGA	F2S_FPGA_IRQ14	-	Level or Edge
66	FPGA	F2S_FPGA_IRQ15	-	Level or Edge
67	FPGA	F2S_FPGA_IRQ16	-	Level or Edge
68	FPGA	F2S_FPGA_IRQ17	-	Level or Edge
69	FPGA	F2S_FPGA_IRQ18	-	Level or Edge

GIC Interrupt Number	Source Block	Interrupt Name	Combined Interrupts	Triggering
70	FPGA	F2S_FPGA_IRQ19	-	Level or Edge
71	FPGA	F2S_FPGA_IRQ20	-	Level or Edge
72	FPGA	F2S_FPGA_IRQ21	-	Level or Edge
73	FPGA	F2S_FPGA_IRQ22	-	Level or Edge
74	FPGA	F2S_FPGA_IRQ23	-	Level or Edge
75	FPGA	F2S_FPGA_IRQ24	-	Level or Edge
76	FPGA	F2S_FPGA_IRQ25	-	Level or Edge
77	FPGA	F2S_FPGA_IRQ26	-	Level or Edge
78	FPGA	F2S_FPGA_IRQ27	-	Level or Edge
79	FPGA	F2S_FPGA_IRQ28	-	Level or Edge
80	FPGA	F2S_FPGA_IRQ29	-	Level or Edge
81	FPGA	F2S_FPGA_IRQ30	-	Level or Edge
82	FPGA	F2S_FPGA_IRQ31	-	Level or Edge
83	FPGA	F2S_FPGA_IRQ32	-	Level or Edge

GIC Interrupt Number	Source Block	Interrupt Name	Combined Interrupts	Triggering
84	FPGA	F2S_FPGA_IRQ33	-	Level or Edge
85	FPGA	F2S_FPGA_IRQ34	-	Level or Edge
86	FPGA	F2S_FPGA_IRQ35	-	Level or Edge
87	FPGA	F2S_FPGA_IRQ36	-	Level or Edge
88	FPGA	F2S_FPGA_IRQ37	-	Level or Edge
89	FPGA	F2S_FPGA_IRQ38	-	Level or Edge
90	FPGA	F2S_FPGA_IRQ39	-	Level or Edge
91	FPGA	F2S_FPGA_IRQ40	-	Level or Edge
92	FPGA	F2S_FPGA_IRQ41	-	Level or Edge
93	FPGA	F2S_FPGA_IRQ42	-	Level or Edge
94	FPGA	F2S_FPGA_IRQ43	-	Level or Edge
95	FPGA	F2S_FPGA_IRQ44	-	Level or Edge
96	FPGA	F2S_FPGA_IRQ45	-	Level or Edge
97	FPGA	F2S_FPGA_IRQ46	-	Level or Edge

GIC Interrupt Number	Source Block	Interrupt Name	Combined Interrupts	Triggering
98	FPGA	F2S_FPGA_IRQ47	-	Level or Edge
99	FPGA	F2S_FPGA_IRQ48	-	Level or Edge
100	FPGA	F2S_FPGA_IRQ49	-	Level or Edge
101	FPGA	F2S_FPGA_IRQ50	-	Level or Edge
102	FPGA	F2S_FPGA_IRQ51	-	Level or Edge
103	FPGA	F2S_FPGA_IRQ52	-	Level or Edge
104	FPGA	F2S_FPGA_IRQ53	-	Level or Edge
105	FPGA	F2S_FPGA_IRQ54	-	Level or Edge
106	FPGA	F2S_FPGA_IRQ55	-	Level or Edge
107	FPGA	F2S_FPGA_IRQ56	-	Level or Edge
108	FPGA	F2S_FPGA_IRQ57	-	Level or Edge
109	FPGA	F2S_FPGA_IRQ58	-	Level or Edge
110	FPGA	F2S_FPGA_IRQ59	-	Level or Edge
111	FPGA	F2S_FPGA_IRQ60	-	Level or Edge

GIC Interrupt Number	Source Block	Interrupt Name	Combined Interrupts	Triggering
112	FPGA	F2S_FPGA_IRQ61	-	Level or Edge
113	FPGA	F2S_FPGA_IRQ62	-	Level or Edge
114	FPGA	F2S_FPGA_IRQ63	-	Level or Edge
115	DMA	dma_IRQ0	-	Level
116	DMA	dma_IRQ1	-	Level
117	DMA	dma_IRQ2	-	Level
118	DMA	dma_IRQ3	-	Level
119	DMA	dma_IRQ4	-	Level
120	DMA	dma_IRQ5	-	Level
121	DMA	dma_IRQ6	-	Level
122	DMA	dma_IRQ7	-	Level
123	DMA	dma_irq_abort	-	Level
124	EMAC0	emac0_IRQ	This interrupt combines: sbd_intr_o and lpi_intr_o.	Level
125	EMAC1	emac1_IRQ	This interrupt combines: sbd_intr_o and lpi_intr_o.	Level
126	EMAC2	emac2_IRQ	This interrupt combines: sbd_intr_o and lpi_intr_o.	Level
127	USB0	usb0_IRQ	-	Level
128	USB1	usb1_IRQ	-	Level
129	SDRAM scheduler	HMC_error	-	Level

GIC Interrupt Number	Source Block	Interrupt Name	Combined Interrupts	Triggering
130	SDMMC	sdmmc_IRQ	-	Level
131	NAND	nand_IRQ	-	Level
132	QSPI	qspi_IRQ	-	Level
133	SPI0 master	spim0_IRQ	This interrupt combines: ssi_txe_intr, ssi_txo_intr, ssi_rxf_intr, ssi_rxo_intr, ssi_rxu_intr, and ssi_mst_intr.	Level
134	SPI1 master	spim1_IRQ	This interrupt combines: ssi_txe_intr, ssi_txo_intr, ssi_rxf_intr, ssi_rxo_intr, ssi_rxu_intr, and ssi_mst_intr.	Level
135	SPI0 slave	spis0_IRQ	This interrupt combines: ssi_txe_intr, ssi_txo_intr, ssi_rxf_intr, ssi_rxo_intr, ssi_rxu_intr, and ssi_mst_intr.	Level
136	SPI1 slave	spis1_IRQ	This interrupt combines: ssi_txe_intr, ssi_txo_intr, ssi_rxf_intr, ssi_rxo_intr, ssi_rxu_intr, and ssi_mst_intr.	Level

GIC Interrupt Number	Source Block	Interrupt Name	Combined Interrupts	Triggering
137	I2C0	i2c0_IRQ	This interrupt combines: ic_rx_under_intr, ic_rx_full_intr, ic_tx_over_intr, ic_tx_empty_intr, ic_rd_req_intr, ic_tx_abrt_intr, ic_rx_done_intr, ic_activity_intr, ic_stop_det_intr, ic_start_det_intr, and ic_gen_call_intr.	Level
138	I2C1	i2c1_IRQ	This interrupt combines: ic_rx_under_intr, ic_rx_full_intr, ic_tx_over_intr, ic_tx_empty_intr, ic_rd_req_intr, ic_tx_abrt_intr, ic_rx_done_intr, ic_activity_intr, ic_stop_det_intr, ic_start_det_intr, and ic_gen_call_intr.	Level

GIC Interrupt Number	Source Block	Interrupt Name	Combined Interrupts	Triggering
139	I2C2 (can be used with EMAC0)	i2c2_IRQ	This interrupt combines: ic_rx_under_intr, ic_rx_full_intr, ic_tx_over_intr, ic_tx_empty_intr, ic_rd_req_intr, ic_tx_abrt_intr, ic_rx_done_intr, ic_activity_intr, ic_stop_det_intr, ic_start_det_intr, and ic_gen_call_intr.	Level
140	I2C3 can be used with EMAC1)	i2c3_IRQ	This interrupt combines: ic_rx_under_intr, ic_rx_full_intr, ic_tx_over_intr, ic_tx_empty_intr, ic_rd_req_intr, ic_tx_abrt_intr, ic_rx_done_intr, ic_activity_intr, ic_stop_det_intr, ic_start_det_intr, and ic_gen_call_intr.	Level

GIC Interrupt Number	Source Block	Interrupt Name	Combined Interrupts	Triggering
141	I2C4 (can be used with EMAC2)	i2c4_IRQ	This interrupt combines: ic_rx_under_intr, ic_rx_full_intr, ic_tx_over_intr, ic_tx_empty_intr, ic_rd_req_intr, ic_tx_abrt_intr, ic_rx_done_intr, ic_activity_intr, ic_stop_det_intr, ic_start_det_intr, and ic_gen_call_intr.	Level
142	UART0	uart0_IRQ	-	Level
143	UART1	uart1_IRQ	-	Level
144	GPIO0	gpio0_IRQ	-	Level
145	GPIO1	gpio1_IRQ	-	Level
146	GPIO2	gpio2_IRQ	-	Level
147	Timer0	timer_l4sp_0_IRQ	This interrupt combines: TIMINT1 and TIMINT2.	Level
148	Timer1	timer_l4sp_1_IRQ	This interrupt combines: TIMINT1 and TIMINT2.	Level
149	Timer2	timer_osc1_0_IRQ	This interrupt combines: TIMINT1 and TIMINT2.	Level
150	Timer3	timer_osc1_1_IRQ	This interrupt combines: TIMINT1 and TIMINT2.	Level
151	Watchdog0	wdog0_IRQ	-	Level
152	Watchdog1	wdog1_IRQ	-	Level

GIC Interrupt Number	Source Block	Interrupt Name	Combined Interrupts	Triggering
153	Clock Manager	clkmgr_IRQ	-	Level
154	Reset Manager	restmgr_IRQ	-	Level
155	FPGA Manager	fpga_man_IRQ	-	Level
156	CoreSight	nCTIIRQ[0]	-	Level
157	CoreSight	nCTIIRQ[1]	-	Level
158	Security Manager	SEC_MGR_INTR	-	Level
159	L2-Cache	DATABWERR	-	Edge-triggered

Related Information

Implementation Details on page 9-13

Global Timer

The MPU features a global 64-bit, auto-incrementing timer, which is primarily used by the operating system.

Functional Description

The global timer is accessible by the processors using memory-mapped access through the SCU. The global timer has the following features:

- 64-bit incrementing counter with an auto-incrementing feature. It continues incrementing after sending interrupts.
- Memory-mapped in the private memory region.
- Accessed at reset in Secure State only. It can only be set once, but secure code can read it at any time.
- Accessible to both Cortex-A9 processors in the MPCore.
- Clocked by `mpu_periph_clk`, running at ¼ the rate of `mpu_clk`.

Implementation Details

Each Cortex-A9 processor has a private 64-bit comparator that generates a private interrupt when the counter reaches the specified value. Each Cortex-A9 processor uses the banked ID, ID27, for this interrupt. ID27 is sent to the GIC as a Private Peripheral Interrupt (PPI).

The global timer is clocked by mpu_periph_clk, running at ¼ the rate of mpu_clk.

Related Information

ARM Infocenter

For more information about the global timer, refer to "About the Global Timer" in the *Global timer, Private timers, and Watchdog registers* chapter of the Cortex-A9 MPCore Technical Reference Manual, available on the ARM Infocenter website.

Snoop Control Unit

The SCU manages data traffic for the Cortex-A9 processors and the memory system, including the L2 cache. In a multi-master system, the processors and other masters can operate on shared data. The SCU ensures that each processor operates on the most up-to-date copy of data, maintaining cache coherency.

Functional Description

The SCU is used to connect the Cortex-A9 processors and the ACP to the L2 cache controller. The SCU performs the following functions:

- When the processors are set to SMP mode, the SCU maintains data cache coherency between the processors.

  Note: The SCU does not maintain coherency of the instruction caches.
- Initiates L2 cache memory accesses
- Arbitrates between processors requesting L2 access
- Manages ACP access with cache coherency capabilities.

Related Information

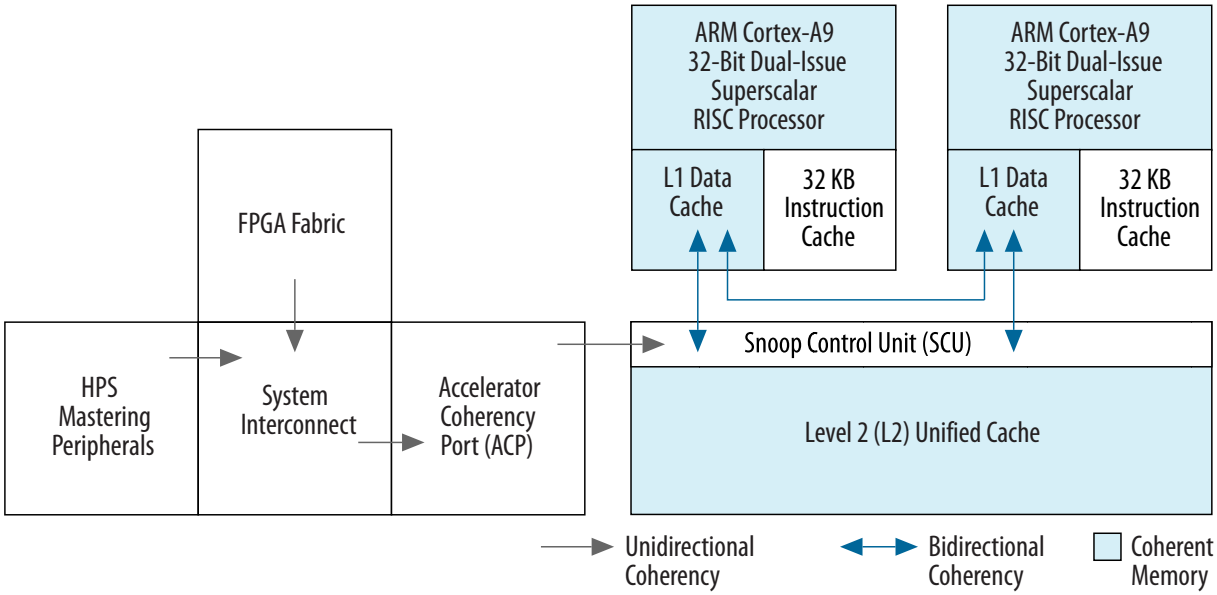
ARM Infocenter

For more information about the SCU, refer to the *Snoop Control Unit* chapter of the *Cortex-A9 MPCore Technical Reference Manual*, available on the ARM Infocenter website.

Coherent Memory, Snoop Control Unit, and Accelerator Coherency Port

Figure 9-3: Data Flow Between L1 Caches and SCU

This diagram illustrates the flow of data among the L1 data caches and the SCU.



Related Information

Accelerator Coherency Port on page 9-29

Implementation Details

When the processor writes to any coherent memory location, the SCU ensures that the relevant data is coherent (updated, tagged or invalidated). Similarly, the SCU monitors read operations from a coherent memory location. If the required data is already stored within the other processor's L1 cache, the data is returned directly to the requesting processor. If the data is not in L1 cache, the SCU issues a read to the L2 cache. If the data is not in the L2 cache memory, the read is finally forwarded to main memory. The primary goal is to maximize overall memory performance and minimize power consumption.

The SCU maintains bidirectional coherency between the L1 data caches belonging to the processors. When one processor performs a cacheable write, if the same location is cached in the other L1 cache, the SCU updates it.

Non-coherent data passes through as a standard read or write operation.

The SCU also arbitrates between the Cortex-A9 processors if both attempt simultaneous access to the L2 cache, and manages accesses from the ACP.

Accelerator Coherency Port

The ACP allows master peripherals—including FPGA-based master peripherals—to maintain data coherency with the Cortex-A9 MPCore processors and the SCU. Dedicated master peripherals in the HPS,

and those built in FPGA logic, access the coherent memory through the ACP. Cacheable master accesses from the system interconnect are redirected to the ACP.

The ACP port allows one-way coherency. One-way coherency allows an external ACP master to see the coherent memory of the Cortex-A9 processors but does not allow the Cortex-A9 processors to see memory changes outside of the cache.

A master on the ACP port can read coherent memory directly from the L1 and L2 caches, but cannot write directly to the L1 cache. The possible ACP master read and write scenarios are as follows:

- ACP master read with coherent data in the L1 cache: The ACP gets data directly from the L1 cache and the read is interleaved with a processor access to the L1 cache.
- ACP master read with coherent data in L2 cache: The ACP request is queued in the SCU and the transaction is sent to the L2 cache.
- ACP master read with coherent data not in L1 or L2 cache: Depending on the previous state of the data, the L1 or L2 cache may request the data from the L3 system interconnect. The ACP is stalled until data is available, however ACP support of multiple outstanding transactions minimizes bottlenecks.
- ACP master write with coherent data in the L1 cache: The L1 cache data is marked as invalid in the Cortex-A9 MPU and the line is evicted from the L1 cache and sent to L2 memory. The ACP write data is scheduled in the SCU and is eventually written to the L2 cache. Later, when the Cortex-A9 processor accesses the same memory location, a cache miss in the L1 occurs.
- ACP master write, with coherent data in the L2 cache: The ACP write is scheduled in the SCU and then is written into the L2 cache.
- ACP master write, with coherent data not in L1 or L2 cache: ACP write is scheduled.

An ACP transaction is generated when a master on the system interconnect has its `AxCACHE[1]` attribute signal set to 1, which indicates a coherent request. All transactions that are set as cacheable are routed to the ACP instead of the normal mapping and are treated as coherent by the cache controllers of the MPU subsystem. The ACP ID generation follows an allocation mechanism that ensures that requests with the same master initiator flow and sequence identifiers always allocate the same local sequence ID and that requests with different identifiers always have different IDs. In addition, if an allocator runs out of sequence IDs, the ACP stalls requests until the reception of responses makes new sequence IDs available. Therefore, the guaranteed number of pending transactions that the interconnect can have is up to four pending transactions, with the allowed AXI ID[2:0] values of 0x4 through 0x7. AxID[2:0] values of 0x0 and 0x1 are assigned to CPU0 and CPU1, respectively.

Note: The entire 4 GB address space can be accessed coherently through the ACP.

Note: Refer to the Arria 10 SoC device errata for details on ARM errata that directly affect the ACP.

Related Information

- **Coherent Memory, Snoop Control Unit, and Accelerator Coherency Port** on page 9-29
- **Arria 10 SX Device Errata**

AxUSER and AxCACHE Attributes

ARUSER[0] and AWUSER[0] Bits

The ACP distinguishes between shared and non-shared requests with the `ARUSER[0]` and `AWUSER[0]` signals. The other AxUSER signals, `AxUSER[4:1]` are not interpreted by the SCU and are simply forwarded to the L2 cache. They are typically set to `AxUSER[4:1]` = b'1111.The `AxCACHE[1]` attribute signals must

also be set to allow L1 caches to be snooped. Sideband signals ARUSER[4:1] and AWUSER[4:1] convey the inner cache attributes.

The table below shows how the AxUSER[0] bit determines whether a request is shared or non-shared. ARUSER[0] applies to read transactions, and AWUSER[0] applies to write transactions.

Table 9-4: ARUSER[0] or AWUSER[0] Sideband Signal Information

ARUSER[0] or AWUSER[0]	Sideband Signal Information
1	Shared request
0	Non-shared request

Other AxUSER bits from the ACP are not interpreted by the SCU but are forwarded to the L2 cache.

ARCACHE[4:0] and AWCACHE[4:0] Bits

The Cortex-A9 MPU only interprets ARCACHE[1] and AWCACHE[1] from ACP requests. All other attributes are forwarded to the L2 cache. AxCACHE[1] is used in combination with AxUSER[0] to detect a coherent access. If AxCACHE[1]=0x1, (normal memory) and AxUSER[0]=0x1, then the access is considered coherent. All ACP requests with AxCACHE[1]=0x0 or AxUSER[0]=0x0 are seen as non-coherent requests.

Table 9-5: ARUSER[0] or AWUSER[0] Sideband Signal Information

ARCACHE[1] or AWCACHE[1]	ARUSER[0] or AWUSER[0]	Access Type
1	1	Coherent request
0	0	Non-coherent request

Cache Coherency for ACP Shared Requests

When a shared access is received on the ACP, caches are checked for coherency of the requested address. A shared access occurs when two masters access the same memory space.

This coherency check is performed by the SCU. If a cache hit occurs during a write access, the affected cache lines are cleaned and invalidated. This event may cause an eviction from the L1 cache to be sent to L2 memory if the L1 cache line is dirty. Once the invalidation and possible eviction is completed, the ACP write request is written to L2 memory. If an eviction was executed from L1 cache, then two consecutive writes to L2 memory occur over the AXI bus: the write from the eviction and the write from the ACP.

For a cache hit during a read access, the L1 cache line is provided by the CPU and is returned to the ACP.

For a cache miss during a write access, the invalidation is considered as complete and the ACP request is sent to L2 memory.

For a cache miss during a read access, the request is forwarded to L2 memory, which returns the data directly to the ACP.

Note: Shared write requests are always transferred to the L2 memory once the cache line is potentially clean and invalidated in the L1 cache memory. Reads are observed on the L2 only if they miss in the L1 cache.

AXI Master Configuration for ACP Access

To use the ACP for coherent accesses, the following configurations apply:

ACP master configurations must be as follows:

- For coherent ACP read accesses, the AXI bits must be programmed as follows to avoid compromising coherency:
  - `AxCACHE[3:0]` attributes must match the properties defined in the Cortex-A9 MPCore MMU page tables for the relevant memory region.
  - Shareable attribute `AxUSER[0]` must be set to 0x1

The Cortex-A9 MPCore configuration for ACP use should be as follows:

- The Snoop Control Unit must be enabled (by setting the SCU enable bit in the SCU Control Register at 0xFFFFC000).
- Coherent memory must be marked cacheable and shareable.
- The `SMP` bit of the `ACTLR` register must be set in the Cortex-A9 processor that shares data over the ACP.

Note: To achieve maximum performance on the ACP, avoid switching from shared to non-shared requests and vice-versa. When a shared request is latched in the ACP and there are non-shared requests still pending, the non-shared requests must be completed before the shared request can proceed.

The following sections detail the attribute configurations necessary to support coherency.

Related Information

Cortex-A9 MPCore Technical Reference Manual
For more information about the use of the L2 Cache Controller address space, refer to the *Cortex-A9 MPCore Technical Reference Manual*, available on the ARM Infocenter website.

Configuring AxCACHE[3:0] Sideband Signals for Coherent Accesses

The following list highlights how to correctly derive and apply the correct `AxCACHE` settings for coherent accesses.

- The correct `AxCACHE[3:0]` setting is dependent on the MMU page table settings. However, for coherent accesses, `AxCACHE[1]` must be set to 0x1.
- For HPS masters, `AxCACHE[3:0]` is static and applied to the relevant master port through the System Manager.
- For FPGA masters, `AxCACHE[3:0]` is applied in the FPGA fabric and can be set for each access.

Related Information

- **System Manager** on page 5-1
  For more information about programming AxCACHE[3:0] through the System Manager
- **Cortex-A9 MPCore Technical Reference Manual**
  For more information about the Accelerator Coherency Port.

Configuring AxUSER[4:0] Sideband Signals

The following list highlights how to correctly derive and apply the correct `AxUSER` settings for coherent accesses.

- AxUSER[0] (shared attribute) must be set to 0x1 for coherent accesses.
- Because the ACP has no inner cache policy, AxUSER[3:1] is not interpreted by the SCU. AxUSER[3:1] attributes pass to the L2 cache controller and are used when the cache is operating in exclusive mode.
- For FPGA masters, AxUSER[4:0] is applied in the FPGA fabric and can be set for each access.

Related Information

Cortex-A9 MPCore Technical Reference Manual
For more information about the Accelerator Coherency Port.

Burst Sizes and Byte Strobes

The ACP improves system performance for hardware accelerators in the FPGA fabric. However, in order to achieve high levels of performance, you must use the one of the optimized burst types. Other burst configurations have significantly lower performance.

Recommended Burst Types

Table 9-6: Recommended Burst Types for Optimized Bursts

Burst Type	Beats	Width (Bits)	Address Type	Byte Strobes
Wrapping	4	64	64-bit aligned	Asserted
Incrementing	4	64	32-bit aligned	Asserted

Note: If the slave port of the FPGA-to-HPS bridge is not 64 bits wide, you must supply bursts to the FPGA-to-HPS bridge that are upsized or downsized to the burst types above. For example, if the slave data width of the FPGA-to-HPS bridge is 32 bits, then bursts of eight beats by 32 bits are required to access the ACP efficiently.

Note: If the address and burst size of the transaction to the ACP matches either of the conditions shown in the table "Recommended Burst Types for Optimized Bursts", the logic in the MPU assumes the transaction has all its byte strobes set. If the byte strobes are not all set, then the write does not actually overwrite all the bytes in the word. Instead, the cache assumes the whole cache line is valid. If this line is dirty (and therefore gets written out to SDRAM), data corruption might occur.

In addition to optimizing performance, using a 64-bit access width will allow you to use ECC. ECC is only supported for 64-bit accesses that are 64-bit aligned..

Related Information

Single Event Upset Protection on page 9-37

Exclusive and Locked Accesses

The ACP does not support exclusive accesses to coherent memory. To ensure mutually exclusive access to shared data, use the exclusive access support built into the SDRAM L3 interconnect scheduler. The AXI buses that interface to the scheduler provide ARLOCK[0] and AWLOCK[0] signals which are used by the

scheduler to arbitrate for exclusive access to a memory location. The SDRAM L3 scheduler contains one monitor for each of the following exclusive-capable masters:

- CPU0
- CPU1
- FPGA-to-HPS
- FPGA-to-SDRAM0
- FPGA-to-SDRAM1
- FPGA-to-SDRAM2

Each monitor only tracks one exclusive address.

Related Information

System Interconnect on page 7-1
To learn more about Exclusive Monitors in the System Interconnect, refer to the System Interconnect Chapter.

L2 Cache

The MPU subsystem includes a secondary 512 KB L2 shared, unified cache memory.

Functional Description

The L2 cache is much larger than the L1 cache. The L2 cache has significantly lower latency than external memory. The L2 cache is up to eight-way associative, configurable down to one-way (direct mapped). Like the L1 cache, the L2 cache can be locked by cache line, locked by way, or locked by master (CPU and ACP masters).

The L2 cache implements error correction codes (ECCs) and ECC error reporting. The cache can report a number of events to the processor and operating system.

Related Information

Cortex-A9 MPU Subsystem Register Implementation on page 9-46
For more information regarding the L2 Cache Controller Address Map, refer to the *Cortex-A9 MPU Subsystem Register Implementation* section.

Cache Controller Configuration

The L2 cache consists of the ARM L2C-310 L2 cache controller configured as follows:

- 512 KB total memory
- Eight-way associativity
- Physically addressed, physically tagged
- Line length of 32 bytes
- Critical first word linefills

- Support for all AXI cache modes
  - Write-through
  - Write-back
  - Read allocate
  - Write allocate
  - Read and write allocate
- Single event upset (SEU) protection
  - Parity on Tag RAM
  - ECC on L2 Data RAM
- Two slave ports mastered by the SCU
- Two master ports connected to the following slave ports:
  - SDRAM L3 interconnect, 64-bit slave port width
  - L3 interconnect, 64-bit slave port width
- Cache lockdown capabilities as follows:
  - Line lockdown
  - Lockdown by way
  - Lockdown by master (both processors and ACP masters)
- TrustZone support
- Cache event monitoring

Related Information

- **L2 Cache Event Monitoring** on page 9-36
- **System Manager** on page 5-1
  For more information about SEU errors, refer to the *System Manager* chapter.
- **ARM Infocenter**
  For more information about AXI user sideband signals, refer to the CoreLink Level 2 Cache Controller L2C-310 Technical Reference Manual, which you can download from the ARM Infocenter website.

Implementation Details

The following table shows the parameter settings for the cache controller.

Table 9-7: Cache Controller Configuration

Feature	Meaning
Cache way size	64 KB
Number of cache ways	8 ways
Tag RAM write latency	1
Tag RAM read latency	1
Tag RAM setup latency	1
Data RAM write latency	1

Feature	Meaning
Data RAM read latency	2
Data RAM setup latency	1
Parity logic	Parity logic enabled
Lockdown by master	Lockdown by master enabled
Lockdown by line	Lockdown by line enabled
AXI ID width on slave ports	6 AXI ID bits on slave ports
Address filtering	Address filtering logic enabled
Speculative read	Logic for supporting speculative read enabled
Presence of ARUSERMx and AWUSERMx sideband signals	Sideband signals enabled

Related Information

ARM Infocenter

For further information about cache controller configurable options, refer to the CoreLink Level 2 Cache Controller L2C-310 Technical Reference Manual, available on the ARM Infocenter website.

L2 Cache Event Monitoring

The L2 cache supports the built-in cache event monitoring signals shown in the table below. The L2 cache can count two of the events at any one time.

Table 9-8: L2 Cache Events

Event	Description
`CO`	Eviction (cast out) of a line from the L2 cache.
`DRHIT`	Data read hit in the L2 cache.
`DRREQ`	Data read lookup to the L2 cache. Subsequently results in a hit or miss.
`DWHIT`	Data write hit in the L2 cache.
`DWREQ`	Data write lookup to the L2 cache. Subsequently results in a hit or miss.

Event	Description
DWTREQ	Data write lookup to the L2 cache with write-through attribute. Subsequently results in a hit or miss.
EPFALLOC	Prefetch hint allocated into the L2 cache.
EPFHIT	Prefetch hint hits in the L2 cache.
EPFRCVDS0	Prefetch hint received by slave port S0.
EPFRCVDS1	Prefetch hint received by slave port S1.
IPFALLOC	Allocation of a prefetch generated by L2 cache controller into the L2 cache.
IRHIT	Instruction read hit in the L2 cache.
IRREQ	Instruction read lookup to the L2 cache. Subsequently results in a hit or miss.
SPNIDEN	Secure privileged non-invasive debug enable.
SRCONFS0	Speculative read confirmed in slave port S0.
SRCONFS1	Speculative read confirmed in slave port S1.
SRRCVDS0	Speculative read received by slave port S0.
SRRCVDS1	Speculative read received by slave port S1.
WA	Allocation into the L2 cache caused by a write (with write-allocate attribute) miss.

In addition, the L2 cache events can be captured and timestamped using dedicated debugging circuitry.

Related Information

ARM Infocenter

For more information about L2 event capture, refer to the *Debug* chapter of the Cortex-A9 MPCore Technical Reference Manual, available on the ARM Infocenter website.

Single Event Upset Protection

The L2 cache has the option of using ECCs to protect against Single Event Upset (SEU) errors in the cache RAM.

The L2 cache has two types of protection:

- The L2 cache tag RAMs are protected by parity.
- The L2 cache data RAMs are protected using single error correction, double error detection (SECDED) ECC Hamming code.

Enabling ECCs does not affect the performance of the L2 cache. The ECC bits are calculated only for writes to the data RAM that are 64 bits wide (8 bytes, or one-quarter of the cache line length). The ECC logic does not perform a read-modify-write when calculating the ECC bits.

The ECC protection bits are not valid in the following cases:

- Data is written that is not 64-bit aligned in memory
- Data is written that is less than 64 bits in width

In these cases the Byte Write Error interrupt is asserted. Cache data is still written when such an error occurs. However, the ECC error detection and correction continues to function. Therefore, the cache data is likely to be incorrect on subsequent reads.

To use ECCs, the software and system must meet the following requirements:

- L1 and L2 cache must be configured as write-back and write-allocate for any cacheable memory region
- Level 3 interconnect masters using the ACP must only perform the following types of data writes:
  - 64-bit aligned in memory
  - 64-bit wide accesses

  Note that system interconnect masters can include masters in the FPGA accessing the FPGA-to-HPS bridge.

If a correctable ECC error occurs, the ECC corrects the read data in parallel to asserting a correctable error signal on the AXI bus. If an uncorrectable error occurs in the L2 cache, the uncorrected data remains in the L2 cache and an AXI slave error (SLVERR) is sent to the L1 memory system. In addition, interrupts can be enabled for correctable and uncorrectable ECC errors through the GIC.

Related Information

System Manager on page 5-1
For more information about SEU errors, refer to the *System Manager* chapter.

L2 Cache Parity

Because the L2 data RAM is ECC-protected, parity checking on the data RAM is not required. However, because the tag RAM is not ECC protected, it requires parity checking. Because parity cannot be configured independently, parity checking for both the data RAM and tag RAM must be enabled.

Ideally, parity should be enabled before the L2 cache is enabled. If the cache is already enabled, you must clean the cache and disable it before parity is enabled. After enabling parity, the cache is invalidated and enabled. To enable parity:

1. Set the `Parity_on` bit in the L2 `Auxiliary Control register`.
2. Invalidate the L2 cache through the `Cache Maintenance Operations` registers.
3. Enable the cache through the `L2_Cache_enable bit` of the L2 `Control register`.

Note: If you would like the parity errors to be reported you must enable the parity interrupts in the `L2 Interrupt Mask` register along with the corresponding interrupts in the general interrupt controller (GIC).

Refer to the ARM Infocenter website for more information regarding L2 cache registers and programming.

Parity Error Handling

If a parity error occurs in the tag or data RAM during AXI read transactions, a SLVERR response is reported through the event bus and interrupt signals. If a parity error occurs on tag or data RAM during AXI write transactions, a SLVERR response is reported back through a SLVERRINTR interrupt signal. For cache maintenance operations, if a parity error occurs on tag or data RAM, the error is reported back through the interrupt lines only. In addition, the L2 Cache cannot refill a line from external memory due to a parity error.

Related Information

ARM Infocenter

L2 Cache Lockdown Capabilities

The L2 cache has three methods to lock data in the cache RAMs:

- Lockdown by line—Used to lock lines in the cache. This is commonly used for loading critical sections of software into the cache temporarily.
- Lockdown by way—Allows any or all of the eight cache ways to be locked. This is commonly used for loading critical data or code into the cache.
- Lockdown by master-- allows cache ways to be dedicated to a single master port. This allows a large cache to look like smaller caches to multiple master ports. The L2 cache can be mastered by any of the L3 masters (including the FPGA fabric). When using lockdown by master, AXI ID0 and 1 represent CPU0 and CPU1 respectively and AXI IDs 0x4 through 0x7 can be used for the L3 interconnect masters.

Related Information

ARM Infocenter

For more information about L2 cache lockdown capabilities, refer to "Cache operation" in the *Functional Overview* chapter of the *CoreLink Level 2 Cache Controller L2C-310 Technical Reference Manual*, available on the ARM Infocenter website.

L2 Cache Event Monitoring

The L2 cache supports the built-in cache event monitoring signals shown in the table below. The L2 cache can count two of the events at any one time.

Table 9-9: L2 Cache Events

Event	Description
`CO`	Eviction (cast out) of a line from the L2 cache.
`DRHIT`	Data read hit in the L2 cache.

Event	Description
DRREQ	Data read lookup to the L2 cache. Subsequently results in a hit or miss.
DWHIT	Data write hit in the L2 cache.
DWREQ	Data write lookup to the L2 cache. Subsequently results in a hit or miss.
DWTREQ	Data write lookup to the L2 cache with write-through attribute. Subsequently results in a hit or miss.
EPFALLOC	Prefetch hint allocated into the L2 cache.
EPFHIT	Prefetch hint hits in the L2 cache.
EPFRCVDS0	Prefetch hint received by slave port S0.
EPFRCVDS1	Prefetch hint received by slave port S1.
IPFALLOC	Allocation of a prefetch generated by L2 cache controller into the L2 cache.
IRHIT	Instruction read hit in the L2 cache.
IRREQ	Instruction read lookup to the L2 cache. Subsequently results in a hit or miss.
SPNIDEN	Secure privileged non-invasive debug enable.
SRCONFS0	Speculative read confirmed in slave port S0.
SRCONFS1	Speculative read confirmed in slave port S1.
SRRCVDS0	Speculative read received by slave port S0.
SRRCVDS1	Speculative read received by slave port S1.
WA	Allocation into the L2 cache caused by a write (with write-allocate attribute) miss.

In addition, the L2 cache events can be captured and timestamped using dedicated debugging circuitry.

Related Information

ARM Infocenter
For more information about L2 event capture, refer to the *Debug* chapter of the Cortex-A9 MPCore Technical Reference Manual, available on the ARM Infocenter website.

L2 Cache Address Filtering

The L2 cache can access either the system interconnect fabric or the SDRAM L3 interconnect. The L2 cache address filtering determines how much address space is allocated to the HPS-to-FPGA bridge and how much is allocated to SDRAM, depending on the configuration of the memory management unit.

Related Information

- **Cortex-A9 MPU Subsystem with System Interconnect** on page 9-2
- **Memory Management Unit** on page 9-10
  Information about how the address space is set based on L2 cache address filtering.

Cache Latency

Latency for cache hits and misses varies.

The latency for an L1 cache hit is 1 clock. The latency for an L1 cache miss and L2 cache hit is 6 clocks best case. Latency in the L2 cache can vary depending on other operations in the L2. Parity and ECC settings have no effect on latency. A single-bit ECC error is corrected during the L2 read, but is not re-written to the L2 RAM.

CPU Prefetch

The Cortex-A9 performs instruction and data prefetches regardless of the state of the MMU.

A prefetch occurs when any address that is 4 KB above the current instruction pointer is accessed. The system has been designed to ensure that the system bus does not lock on prefetches. Any prefetches to unmapped memory space produce a decode error on the system interconnect.

TrustZone

As platforms allow more outside application downloads and as sensitive data is shared on devices, risks of hardware or data asset attacks rise. The Cortex-A9 MPU subsystem has integrated TrustZone technology which provides a system solution to protect application platforms from malicious attack. The TrustZone hardware and supporting software are designed to provide a strong security solution regardless of the operating environment. TrustZone creates a separation between the secure and non-secure areas of the SoC and allows the designer to choose which assets in a design are designated as secure and non-secure.

TrustZone security is implemented in the Cortex-A9 MPU subsystem in three ways:

- Hardware partitioning: Resources can be assigned and identified as secure or non-secure.
- Virtual processor execution: Each core can context switch between secure and non-secure operation.
- Secure debug control: The MPU subsystem provides both secure and non-secure hardware debug features. The type of debug allowed can be configured by the user.

Related Information

TrustZone
For more information about the use of the TrustZone, refer to the *Cortex-A9 MPCore Technical Reference Manual*, available on the ARM Infocenter website.

Secure Partitioning

System interconnect and cache accesses as well as processor execution can be securely partitioned using the TrustZone infrastructure.

Secure Bus Accesses

The Cortex-A9 MPCore and other masters that utilize the system interconnect can be configured for secure or non-secure accesses.

Master accesses drive a protection signal, `AxPROT[1]`, to communicate the security level of the attempted transaction. The Cortex-A9 drives this signal low for secure accesses and high for non-secure accesses. The secure firewalls determine whether the access is allowed or not. A slave access type defaults to secure if no other configuration is programmed. Users can define whether a bus error is generated or if a bus failure occurs when a secure access is violated.

Note: Secure processor configuration is programmed through the `CP15` register.

Related Information

- **Arria 10 HPS Secure Firewalls** on page 6-24
  For more detailed information about secure bus accesses, refer to the "Secure Firewall" section.
- **SoC Security** on page 6-1
  For more information about SoC Security, refer to the *SoC Security* Chapter.

Secure Cache Accesses

Secure and non-secure partitioning also extends to the L1 and L2 caches.

There is a 32-bit physical address space for secure and non-secure transactions and a 33rd bit is provided to indicate security. The cache is able to store both secure and non-secure lines.

The Accelerator Coherency Port (ACP) in the ARM Cortex-A9 MPCore can be used with TrustZone technology. The security state of the masters using the ACP must be the same as the Cortex-A9 processor.

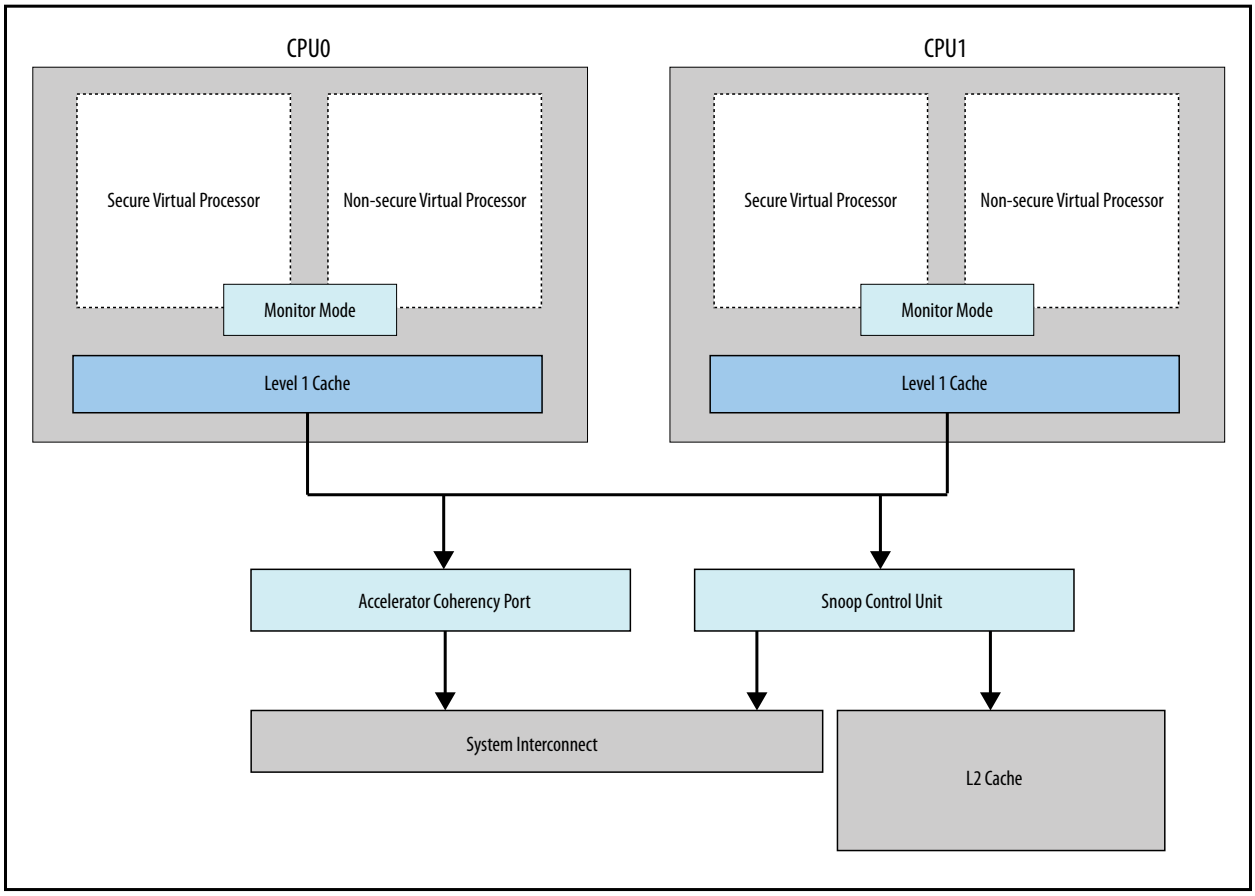
Related Information

- **ARM TrustZone**
  Refer to this location for information about virtual core operation.
- **SoC Security** on page 6-1
  For more information about SoC Security, refer to the *SoC Security* Chapter.

Virtual Processor Operation

Two virtual processors (secure and non-secure) with context switch capability (monitor mode) exist for each Cortex-A9 processor core. The secure virtual processor only accesses secure resources and the non-secure virtual processor only accesses non-secure resources.

Figure 9-4: Virtual Processor Environment with Monitor Mode



A context switch to secure operation is achieved through a secure monitor call (SMC) instruction or the following hardware exceptions (if configured to do so):

- IRQ interrupt
- Faster Interrupt Request (FIQ) interrupt
- External data abort
- External prefetch abort

When a context switch occurs, the state of the current mode is saved and restored on the next context switch or on a return from exception.

Exception Vector Tables

Three exception vector tables are provided for the MPU with TrustZone:

1. Non-secure
2. Secure
3. Monitor mode

Typically IRQs are used for non-secure interrupts and FIQs are used for secure interrupts. The location of each of the three vector tables can be moved at runtime.

The Generic Interrupt Controller (GIC) can handle secure and non-secure interrupts and prevent non-secure accesses from reading or modifying the configuration of a secure interrupt. An interrupt can be made secure by programming the appropriate bits in the Interrupt Security Register. Secure interrupts are always given a higher priority than non-secure interrupts.

Secure Debug

Debug security extensions exist for the MPU subsystem and the system SoC. Secure privileged and user processor debug access can be controlled by two input pins to the core logic. Depending on the configuration of these pins, secure privileged or user JTAG or trace can be used. In addition, normal user debug visibility can be allowed while preventing all secure world debug. To disable all visibility to the core, both secure and non-secure, a global debug pin, `NIDEN`, is provided to the core. Each Cortex-A9 processor in an MPU subsystem has its own dedicated debug visibility signals.

Related Information

CoreSight Debug and Trace on page 10-1

Debugging Modules

The MPU subsystem includes debugging resources through ARM CoreSight on-chip debugging and trace. The following functionality is included:

- Individual program trace for each processor
- Event trace for the Cortex-A9 MPCore
- Cross triggering between processors and other HPS debugging features

Program Trace

Each processor has an independent program trace monitor (PTM) that provides real-time instruction flow trace. The PTM is compatible with a number of third-party debugging tools.

The PTM provides trace data in a highly compressed format. The trace data includes tags for specific points in the program execution flow, called waypoints. Waypoints are specific events or changes in the program flow.

The PTM recognizes and tags the waypoints listed in **Table 9-10**.

Table 9-10: Waypoints Supported by the PTM

Type	Additional Waypoint Information
Indirect branches	Target address and condition code
Direct branches	Condition code
Instruction barrier instructions	—
Exceptions	Location where the exception occurred
Changes in processor instruction set state	—
Changes in processor security state	—
Context ID changes	—
Entry to and return from debug state when Halting debug mode is enabled	—

The PTM optionally provides additional information for waypoints, including the following:

- Processor cycle count between waypoints
- Global timestamp values
- Target addresses for direct branches

Related Information

- **CoreSight Debug and Trace** on page 10-1
- **ARM Infocenter**
  For more information about the PTM, refer to the CoreSight PTM-A9 Technical Reference Manual, available on the ARM Infocenter website.

Event Trace

Events from each processor can be used as inputs to the PTM. The PTM can use these events as trace and trigger conditions.

Related Information

- **Performance Monitoring Unit** on page 9-12
- **ARM Infocenter**
  For more information about the trigger and trace capabilities, refer to the CoreSight PTM-A9 Technical Reference Manual, Revision r1p0, available on the ARM Infocenter website.

Cross-Triggering

The PTM can export trigger events and perform actions on trigger inputs. The cross-trigger signals interface with other HPS debugging components including the FPGA fabric. Also, a breakpoint in one processor can trigger a break in the other.

Related Information

CoreSight Debug and Trace on page 10-1
For detailed information about cross-triggering and about debugging hardware in the MPU, refer to the *CoreSight Debug and Trace* chapter.

Clocks

Three synchronous clocks and two debug clocks are provided to the MPU subsystem.

Table 9-11: MPU Subsystem Clocks

System Clock Name	Use
`mpu_clk`	Main clock for the MPU subsystem
`mpu_periph_clk`	Clock for peripherals inside the MPU subsystem
`mpu_l2_ram_clk`	Clock for L2 cache and AXI interface clocks to and from the system interconnect
`dbg_at_clk`	Trace bus clock
`dbg_clk`	Debug clock

Related Information

Clock Manager on page 2-1

Cortex-A9 MPU Subsystem Register Implementation

The following configurations are available through registers in the Cortex-A9 subsystem:

- All processor-related controls, including the MMU and L1 caches, are controlled using the Coprocessor 15 (CP15) registers of each individual processor.
- All SCU registers, including control for the timers and GIC, are memory mapped.
- All L2 cache registers are memory-mapped.

Related Information

ARM Infocenter
For detailed definitions of the registers for the Altera Cortex-A9 MPU subsystem, refer to the Cortex-A9 MPCore Technical Reference Manual, Revision r4p1 and the CoreLink Level 2 Cache Controller L2C-310 Technical Reference Manual, Revision r3p3, available on the ARM Infocenter website.

Cortex-A9 MPU Subsystem Address Map for Arria 10

The Cortex-A9 MPU subsystem registers reside within the address range from 0xFFFFC000 to 0xFFFFEFFF.

mpu_regs_MPUSCU Address Map

This address space is allocated for the Snoop Control Unit (SCU) registers. For detailed information about the use of this address space, go to the **ARM Infocenter** to access the ARM documentation for the Cortex-A9 MPCore.

Module Instance	Base Address	End Address
mpu_regs_MPUSCU	0xFFFFC000	0xFFFFC0FF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

mpu_regs_MPUGIC Address Map

This address space is allocated for the General Interrupt Controller (GIC) registers. For detailed information about the use of this address space, go to the **ARM Infocenter** to access the ARM documentation for the Cortex-A9 MPCore.

Module Instance	Base Address	End Address
mpu_regs_MPUGIC	0xFFFFC100	0xFFFFC1FF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

mpu_regs_MPUGLOBALTIMER Address Map

This address space is allocated for the Global Timer registers. For detailed information about the use of this address space, go to the **ARM Infocenter** to access the ARM documentation for the Cortex-A9 MPCore.

Module Instance	Base Address	End Address
mpu_regs_MPUGLOBALTIMER	0xFFFFC200	0xFFFFC5FF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

mpu_regs_MPUPRIVATETIMER Address Map

This address space is allocated for the Private Timers and Watchdog registers. For detailed information about the use of this address space, go to the **ARM Infocenter** to access the ARM documentation for the Cortex-A9 MPCore.

Module Instance	Base Address	End Address
mpu_regs_MPUPRIVATETIMER	0xFFFFC600	0xFFFFCFFF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

mpu_regs_MPUINTRDIST Address Map

This address space is allocated for the Interrupt Distributor registers. For detailed information about the use of this address space, go to the **ARM Infocenter** to access the ARM documentation for the Cortex-A9 MPCore.

Module Instance	Base Address	End Address
mpu_regs_MPUINTRDIST	0xFFFFD000	0xFFFFEFFF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

L2 Cache Controller Address Map for Arria 10

The register space for the L2 cache controller ranges from 0xFFFFF000 to 0xFFFFFFFF

l2_regs_L2TYPE Address Map

This address space is allocated for the Cache ID and Cache Type registers. For detailed information about the use of this address space, go to the **ARM Infocenter** to access the ARM documentation for the L2C-301.

Module Instance	Base Address	End Address
l2_regs_L2TYPE	0xFFFFF000	0xFFFFF0FF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

l2_regs_L2CONTROL Address Map

This address space is allocated for the Control registers. For detailed information about the use of this address space, go to the **ARM Infocenter** to access the ARM documentation for the L2C-301.

Module Instance	Base Address	End Address
l2_regs_L2CONTROL	0xFFFFF100	0xFFFFF1FF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

l2_regs_L2INTERRUPT Address Map

This address space is allocated for the Interrupt and Counter Control registers. For detailed information about the use of this address space, go to the **ARM Infocenter** to access the ARM documentation for the L2C-301.

Module Instance	Base Address	End Address
l2_regs_L2INTERRUPT	0xFFFFF200	0xFFFFF6FF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

l2_regs_L2MAINTENANCE Address Map

This address space is allocated for the Cache Maintenance registers. For detailed information about the use of this address space, go to the **ARM Infocenter** to access the ARM documentation for the L2C-301.

Module Instance	Base Address	End Address
l2_regs_L2MAINTENANCE	0xFFFFF700	0xFFFFF8FF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

l2_regs_L2LOCKDOWN Address Map

This address space is allocated for the Cache Lockdown registers. For detailed information about the use of this address space, go to the **ARM Infocenter** to access the ARM documentation for the L2C-301.

Module Instance	Base Address	End Address
l2_regs_L2LOCKDOWN	0xFFFFF900	0xFFFFFBFF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

l2_regs_L2ADDRESSFILTER Address Map

This address space is allocated for the Address Filtering registers. For detailed information about the use of this address space, go to the **ARM Infocenter** to access the ARM documentation for the L2C-301.

Module Instance	Base Address	End Address
l2_regs_L2ADDRESSFILTER	0xFFFFFC00	0xFFFFFEFF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

l2_regs_L2DEBUG Address Map

This address space is allocated for the Debug, Prefecth and Power registers. For detailed information about the use of this address space, go to the **ARM Infocenter** to access the ARM documentation for the L2C-301.

Module Instance	Base Address	End Address
l2_regs_L2DEBUG	0xFFFFFF00	0xFFFFFFFF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Document Revision History

Date	Version	Changes
October 2016	2016.10.28	• Added "Configuring AxCACHE[3:0] Sideband Signals" and "Configuring AxUser[4:0] Sideband Signals" subsections to the "AXI Master configuration for ACP Access" section
May 2016	2016.05.27	Maintenance release
May 2016	2016.05.03	Maintenance release
November 2015	2015.11.02	• Reordered "L2 Cache" subsections • Renamed "ECC Support" L2 subsection to be "Single Event Upset Protection" • Added "L2 Cache Parity" subsection in "L2 Cache" section
May 2015	2015.05.04	• Corrected allowed AxID values in "Accelerator Coherency Port" section • Added address maps for the Cortex-A9 MPU subsystem and the L2 cache controller

Date	Version	Changes
December 2014	2014.12.15	• Added bus transaction scenarios in the "Accelerator Coherency Port" section • Added the "AxUSER and AxCACHE" subsection to the "Accelerator Coherency Port" section • Added the "Shared Requests on ACP" subsection to the "Accelerator Coherency Port" section • Added the "Configuration for ACP Use" subsection to the "Accelerator Coherency Port" section • Added parity error handling information to the "L1 Caches" section and the "Cache Controller Configuration" topic of the "L2 Cache" section.
August 2014	2014.08.18	Initial Release

CoreSight Debug and Trace 10

2016.10.28

a10_5v4 Subscribe Send Feedback

CoreSight systems provide all the infrastructure you require to debug, monitor, and optimize the performance of a complete HPS design. CoreSight technology addresses the requirement for a multicore debug and trace solution with high bandwidth for whole systems beyond the processor core.

CoreSight technology provides the following features:

- Cross-trigger support between SoC subsystems
- High data compression
- Multi-source trace in a single stream
- Standard programming models for standard tool support

The hard processor system (HPS) debug infrastructure provides visibility and control of the HPS modules, the ARM Cortex-A9 microprocessor unit (MPU) subsystem, and user logic implemented in the FPGA fabric. The debug system design incorporates ARM CoreSight components.

Details of the ARM CoreSight debug components can be viewed by clicking on the following related information links:

Related Information

- **Debug Access Port** on page 10-4
- **System Trace Macrocell** on page 10-5
- **Trace Funnel** on page 10-5
- **Embedded Trace FIFO** on page 10-6
- **AMBA Trace Bus Replicator** on page 10-7
- **Embedded Trace Router** on page 10-6
- **Trace Port Interface Unit** on page 10-7
- **Embedded Cross Trigger System** on page 10-7
- **Program Trace Macrocell** on page 10-12

Features of CoreSight Debug and Trace

The CoreSight debug and trace system offers the following features:

- Real-time program flow instruction trace through a separate Program Trace Macrocell (PTM) for each processor
- Host debugger JTAG interface
- Connections for cross-trigger and STM-to-FPGA interfaces, which enable soft IP generation of triggers and system trace messages
- External trace interface through Trace Port Interface Unit (TPIU) for trace analysis tools
- Custom message injection through the System Trace Macrocell (STM) into the trace stream for delivery to a host debugger
- STM, PTM, and level 3 (L3) trace sources multiplexed into a single stream through the Trace Funnel
- Capability to route trace data to any slave accessible to the Embedded Trace Router (ETR) AXI master connected to the L3 interconnect
- Capability for the following components to trigger each other through the embedded cross-trigger system:
  - Cortex-A9 PTM-0
  - Cortex-A9 PTM-1
  - STM
  - Embedded Trace FIFO (ETF)
  - ETR
  - TPIU
  - Cross Trigger Interface (CTI)
  - FPGA-CTI
  - Cross Trigger Matrix (CTM)

Related Information

Cross Trigger Interface on page 10-9

ARM CoreSight Documentation

The following ARM CoreSight specifications and documentation provide a more thorough description of the ARM CoreSight components in the HPS debug system:

- *CoreSight Technology System Design Guide* (ARM DGI 0012D)
- *CoreSight Architecture Specification*
- *Embedded Cross Trigger Technical Reference Manual* (ARM DDI 0291A)
- *CoreSight Components Technical Reference Manual* (ARM DDI 0314H)
- *CoreSight System Trace Macrocell Technical Reference Manual* (ARM DDI 0444A)
- *System Trace Macrocell Programmers' Model Architecture Specification* (ARM IHI 0054)
- *CoreSight Trace Memory Controller Technical Reference Manual* (ARM DDI 0461B)

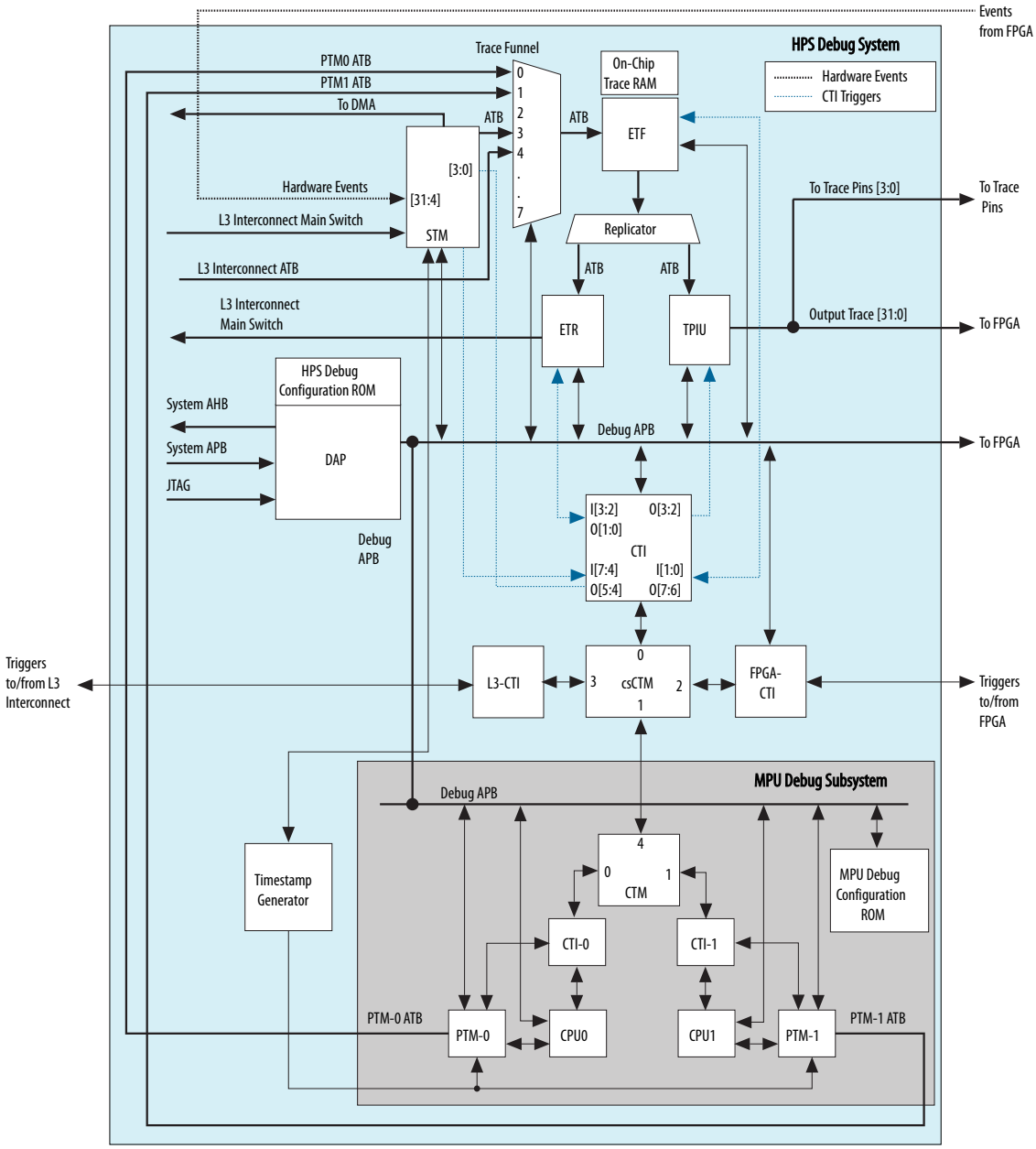
Related Information

ARM Infocenter

You can download these documents from the ARM Infocenter website.

CoreSight Debug and Trace Block Diagram and System Integration

Figure 10-1: HPS CoreSight Debug and Trace System Block Diagram



Functional Description of CoreSight Debug and Trace

Debug Access Port

The Debug Access Port (DAP) provides the necessary ports for a host debugger to connect to and communicate with the HPS through a JTAG interface, which is connected to the FPGA JTAG chain. The JTAG interface provided with the DAP allows a host debugger to access various modules inside the HPS. Additionally, a debug monitor executing on either processor can access different HPS components by interfacing with the system Advanced Microcontroller Bus Architecture (AMBA) Advanced Peripheral Bus (APB) slave port of the DAP.

The system APB slave port occupies 2 MB of address space in the HPS. Both the JTAG port and system APB port have access to the debug APB master port of the DAP.

A host debugger can access any HPS memory-mapped resource in the system through the DAP system master port. Requests made over the DAP system master port are impacted by reads and writes to peripheral registers.

Note: The HPS JTAG interface does not support boundary scan tests (BST).

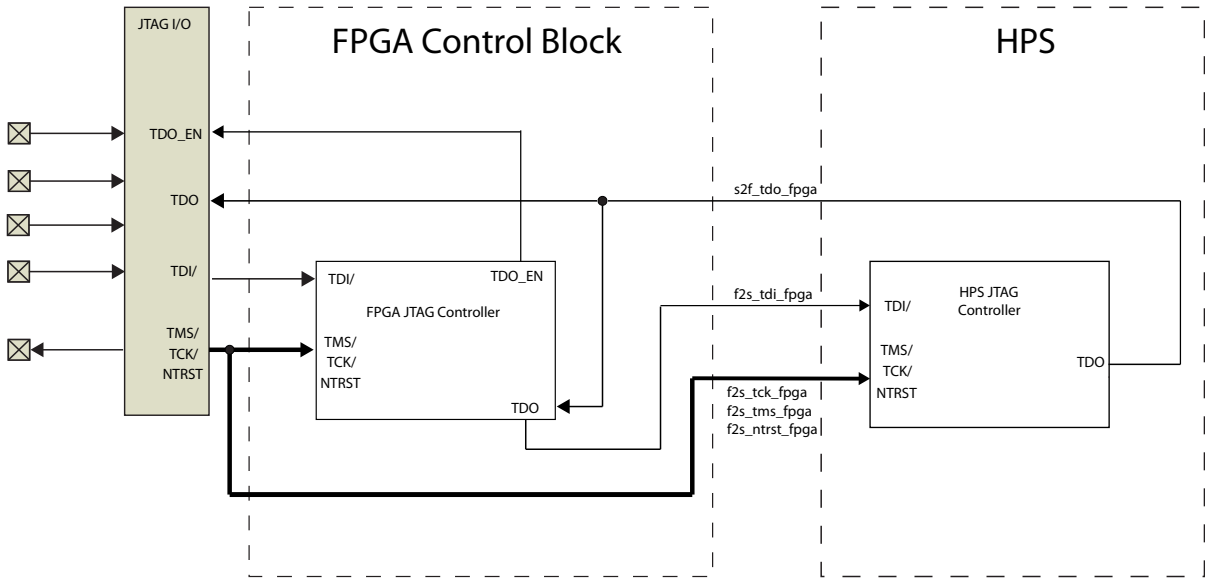
Related Information

- **CoreSight Debug and Trace Block Diagram and System Integration** on page 10-3
  Shows CoreSight components connected to the debug APB
- **Arria 10 Core Fabric and General Purpose I/Os Handbook**
  For more information about boundary scan tests, refer to the "JTAG Boundary-Scan Testing in Arria 10 Devices" chapter.
- **ARM Infocenter**
  Refer to the *CoreSight Components Technical Reference Manual* on the ARM Infocenter website.

JTAG Connection

The SoC device has two JTAG controllers: the FPGA JTAG (located in the configuration sub-system (CSS)) and the HPS JTAG (located in the HPS). The controllers have separate instruction sets and work independently. To share the same external JTAG port between the two controllers, they are internally daisy-chained, where the FPGA appears before the HPS on the chain.

The following figure shows how the two test access port (TAP) controllers are connected:



System Trace Macrocell

The STM allows messages to be injected into the trace stream for delivery to the host debugger receiving the trace data. These messages can be sent through stimulus ports or the hardware EVENT interface. The STM allows these messages to be time stamped.

The STM provides an AMBA Advanced eXtensible Interface (AXI) slave interface used to create trace events. The interface can be accessed by the MPU subsystem, direct memory access (DMA) controller, and masters implemented as soft logic in the FPGA fabric through the FPGA-to-HPS bridge. The AXI slave interface supports three address segments, where each address segment is 16 MB and each segment supports up to 65536 channels. Each channel occupies 256 bytes of address space.

The STM also provides 32 hardware EVENT pins. The higher-order 28 pins (31:4) are connected to the FPGA fabric, allowing logic inside FPGA to insert messages into the trace stream. When the STM detects a rising edge on an EVENT pin, a message identifying the EVENT is inserted into the stream. The lower four EVENT pins (3:0) are connected to CTI.

Related Information

- **HPS-FPGA Bridges** on page 8-1
- **ARM Infocenter**
  For more information, refer to the *CoreSight System Trace Macrocell Technical Reference Manual* on the ARM Infocenter website.
- **CTI** on page 10-21

Trace Funnel

The Trace Funnel is used to combine multiple trace streams into one trace stream. There are four trace streams that use the following funnel ports:

Table 10-1: Trace Stream Connections

Funnel Port	Description
0	The trace stream coming from the PTM connected to CPU0 uses this port.
1	The trace stream coming from the PTM connected to CPU1 uses this port.
2	Not connected.
3	The trace stream coming from the STM uses this port.
4	NoC trace data uses this port.
5 .. 7	Not connected.

Related Information

- **ARM Infocenter**
  Refer to the *CoreSight Components Technical Reference Manual* on the ARM Infocenter website.
- **Program Trace Macrocell** on page 10-12

CoreSight Trace Memory Controller

The CoreSight Trace Memory Controller (TMC) has three possible configurations:

- Embedded Trace FIFO (ETF)
- Embedded Trace Router (ETR)
- Embedded Trace Buffer (ETB)

ETB is not used in this device.

Related Information

ARM Infocenter
For more information, refer to the *CoreSight System Trace Memory Controller Technical Reference Manual* on the ARM Infocenter website.

Embedded Trace FIFO

The Trace Funnel output is sent to the ETF. The ETF is used as an elastic buffer between trace generators (STM, PTM, L3 interconnect) and trace destinations. The ETF stores up to 32 KB of trace data in the on-chip trace RAM.

Related Information

ARM Infocenter
For more information, refer to the *CoreSight System Trace Memory Controller Technical Reference Manual* on the ARM Infocenter website.

Embedded Trace Router

The ETR can route trace data to the HPS on-chip RAM, the HPS SDRAM, and any memory in the FPGA fabric connected to the HPS-to-FPGA bridge. The ETR receives trace data from the AMBA Trace Bus Replicator. By default, the buffer to receive the trace data resides in SDRAM at offset 0x00100000 and is 32 KB. You can override this default configuration by programming registers in the ETR.

Related Information

- **HPS-FPGA Bridges** on page 8-1
- **CoreSight Debug and Trace Programming Model** on page 10-18
- **ARM Infocenter**
  For more information, refer to the *CoreSight System Trace Memory Controller Technical Reference Manual* on the ARM Infocenter website.

AMBA Trace Bus Replicator

The AMBA Trace Bus Replicator broadcasts trace data from the ETF to the ETR and TPIU.

Related Information

ARM Infocenter
Refer to the *CoreSight Components Technical Reference Manual* on the ARM Infocenter website.

Trace Port Interface Unit

The TPIU is a bridge between on-chip trace sources and an off-chip trace port. The TPIU receives trace data from the Trace Bus Replicator (Replicator) and drives the trace data to a trace port analyzer.

The trace output is routed to a 32-bit interface to the FPGA fabric. The trace data sent to the FPGA fabric can be transported off-chip using available serializer/deserializer (SERDES) resources in the FPGA.

Related Information

ARM Infocenter
Refer to the *CoreSight Components Technical Reference Manual* on the ARM Infocenter website.

Embedded Cross Trigger System

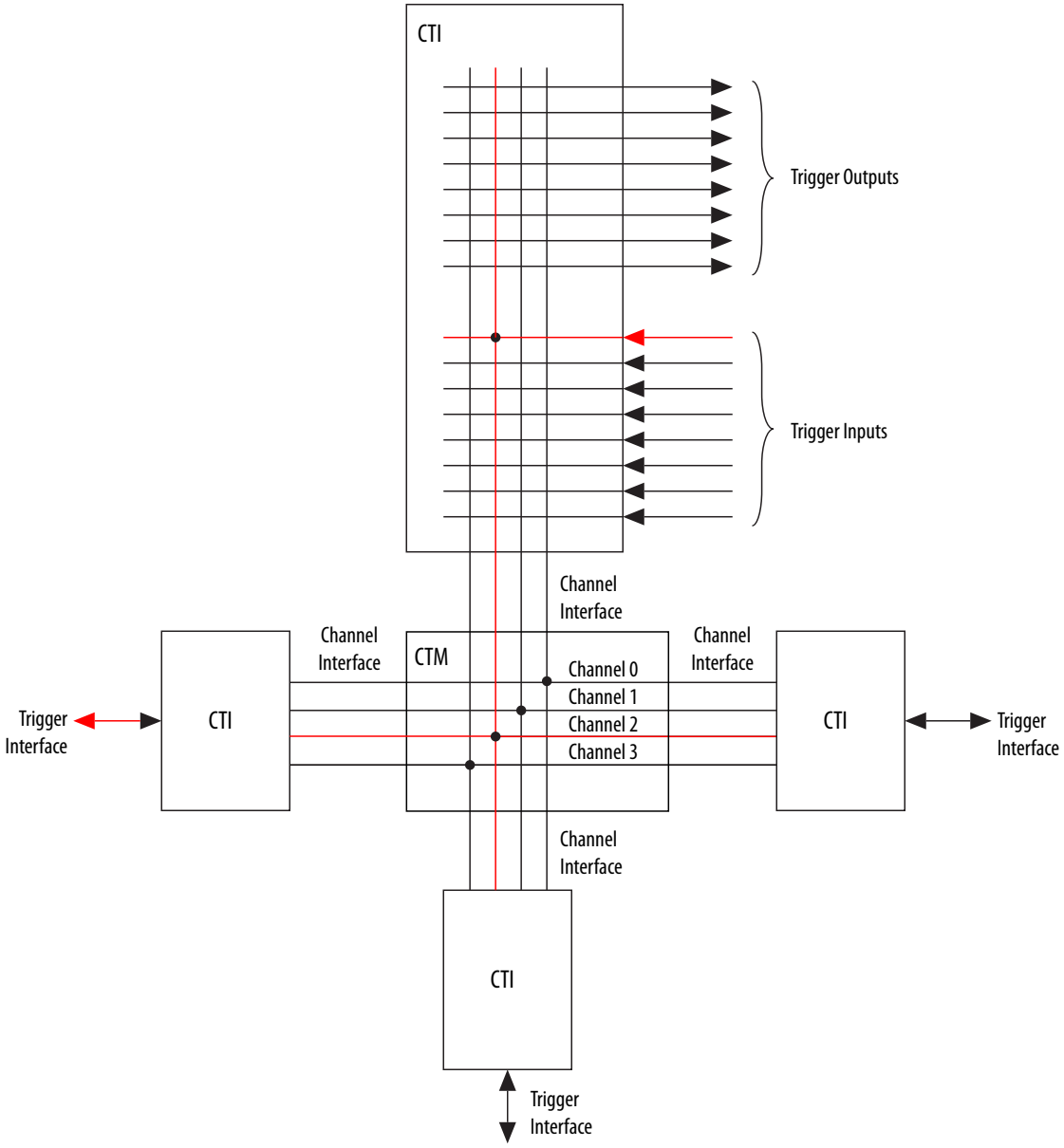
The ECT system provides a mechanism for the components listed in "Features of the CoreSight Debug and Trace" to trigger each other. The ECT consists of the following modules:

- Cross Trigger Interface (CTI)
- Cross Trigger Matrix (CTM)

Figure 10-2: Generic ECT System

The following figure shows an example of how CTIs and CTMs are used in a generic ECT setup. Though the signal travels through channel 2, it only enters and exits through trigger inputs and outputs you configure.

Note: The red line depicts an trigger input to one CTI generating a trigger output in another CTI.



Related Information

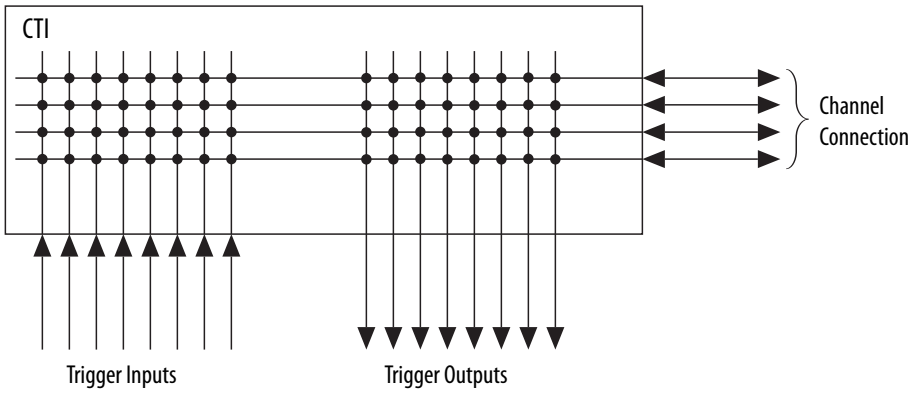
- **Cross Trigger Interface** on page 10-9
- **Features of CoreSight Debug and Trace** on page 10-2
- **Cross Trigger Matrix** on page 10-9

Cross Trigger Interface

CTIs allow trigger sources and sinks to interface with the ECT. Each CTI supports up to eight trigger inputs and eight trigger outputs, and is connected to a CTM. **Figure 10-5** shows the relationship of trigger inputs, trigger outputs, and CTM channels of a CTI.

Figure 10-3: CTI Connections

The following figure shows the eight potential trigger input and trigger output connections that are supported.



The HPS debug system contains the following CTIs:

- CTI—performs cross triggering between the STM, ETF, ETR, and TPIU.
- FPGA-CTI—exposes the cross-triggering system to the FPGA fabric.
- CTI-0 and CTI-1—reside in the MPU debug subsystem. Each CTI is associated with a processor and the processor's PTM.
- L3-CTI—allows cross triggering with L3 interconnect.

Cross Trigger Matrix

A CTM is a transport mechanism for triggers traveling from one CTI to one or more CTIs or CTMs. The HPS contains two CTMs. One CTM connects CTI and FPGA-CTI; the other connects CTI-0 and CTI-1. The two CTMs are connected together, allowing triggers to be transmitted between the MPU debug subsystem, the debug system, and the FPGA fabric.

Each CTM has four ports and each port has four channels. Each CTM port can be connected to a CTI or another CTM.

Figure 10-4: CTM Channel Structure

The following figure shows the structure of a CTM channel. Paths inside the CTM are purely combinatorial.

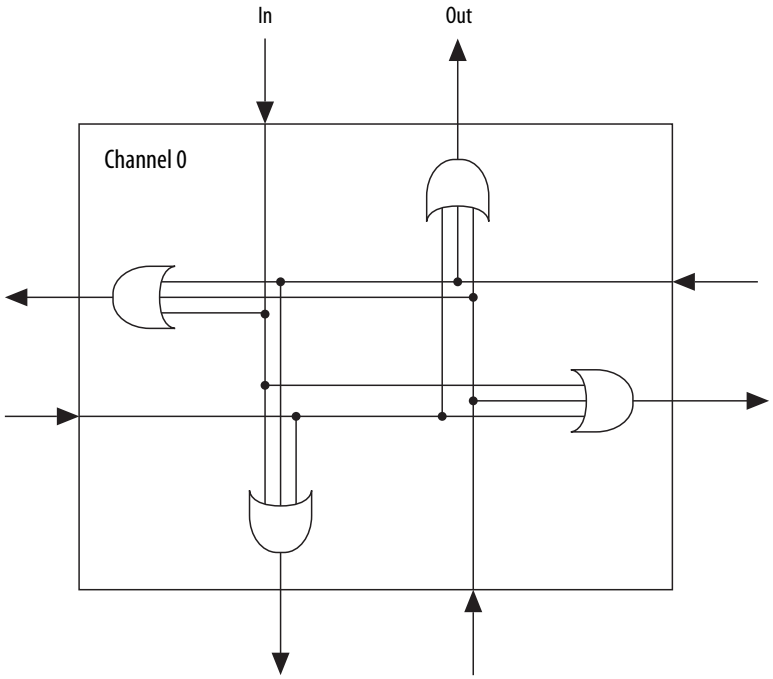
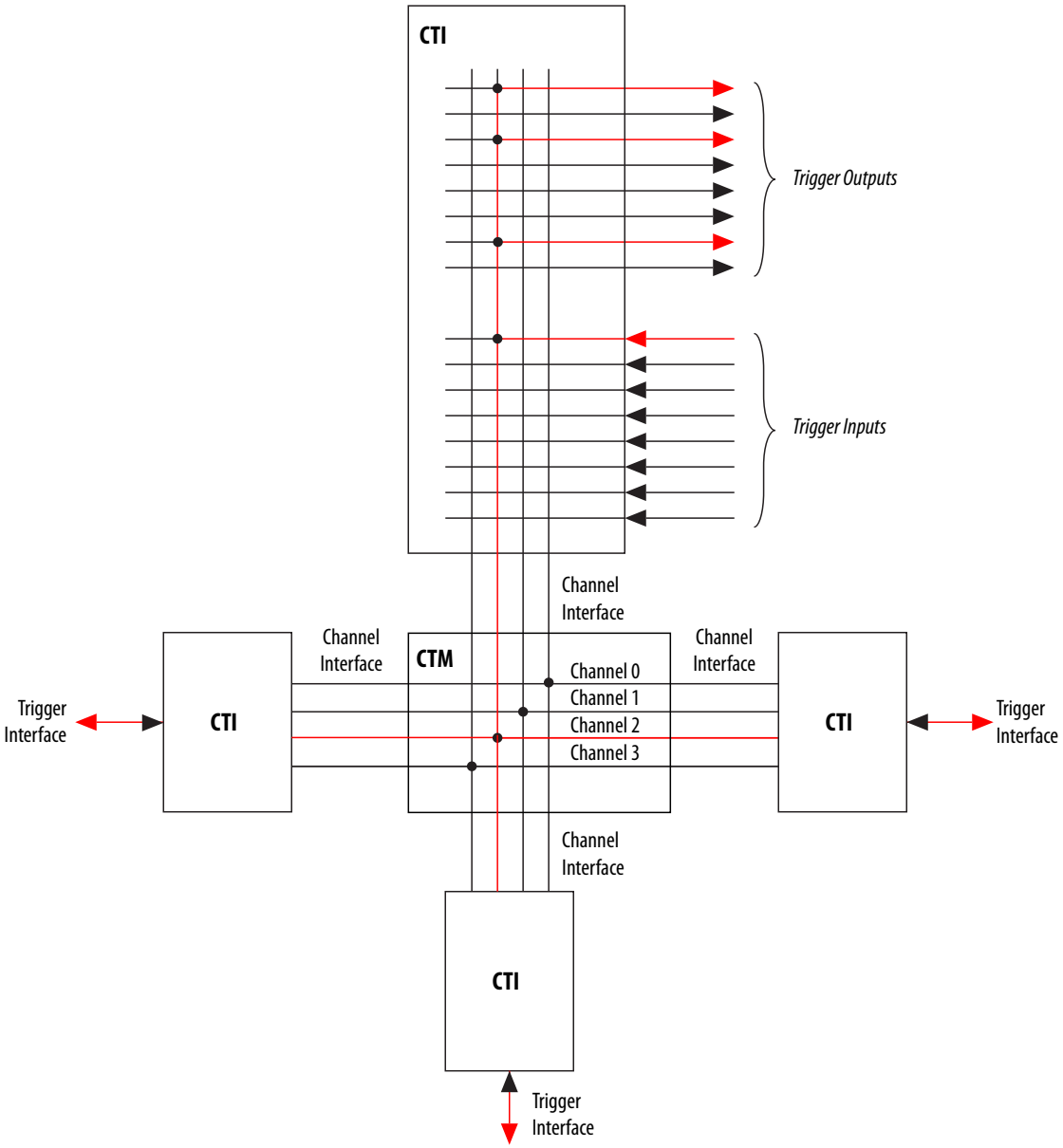


Figure 10-5: CTI Trigger Connections

Each CTI trigger input can be connected through a CTM to one or more trigger outputs under control by a debugger. The following figure shows an example of CTI trigger connections.

Note: The red lines depict the impact one trigger input can have on the entire system.



Related Information

ARM Infocenter

Refer to the *CoreSight Components Technical Reference Manual* on the ARM Infocenter website.

Program Trace Macrocell

The PTM performs real-time program flow instruction tracing and provides a variety of filters and triggers that can be used to trace specific portions of code.

The HPS contains two PTMs. Each PTM is paired with a processor and CTI. Trace data generated from the PTM can be transmitted off-chip using HPS pins, or to the FPGA fabric, where it can be pre-processed and transmitted off-chip using high-speed FPGA pins.

Related Information

ARM Infocenter
For more information, refer to the *CoreSight PTM-A9 Technical Reference Manual* .

HPS Debug APB Interface

The HPS can extend the CoreSight debug control bus into the FPGA fabric. The debug interface is an APB-compatible interface with built-in clock crossing.

Related Information

- **FPGA Interface** on page 10-12
- **HPS Component Interfaces** on page 28-1

FPGA Interface

The following components connect to the FPGA fabric. This section lists the signals from the debug system to the FPGA.

DAP

The DAP uses the system APB port to connect to the FPGA.

Table 10-2: DAP

The following table shows the signal description between DAP and FPGA.

Signal	Description
h2f_dbg_apb_PADDR	Address bus to system APB port, when PADDR
h2f_dbg_apb_PADDR31	Address bus to system APB port, when PADDR31
h2f_dbg_apb_PENABLE	Enable signal from system APB port
h2f_dbg_apb_PRDATA[32]	32-bit system APB port read data bus
h2f_dbg_apb_PREADY	Ready signal to system APB port
h2f_dbg_apb_PSEL	Select signal from system APB port
h2f_dbg_apb_PSLVERR	Error signal to system APB port
h2f_dbg_apb_PWDATA[32]	32-bit system APB port write data bus

Signal	Description
h2f_dbg_apb_PWRITE	Select whether read or write to system APB port • 0 - System APB port read from DAP • 1 - System APB Port write to DAP

STM

The STM has 28 event pins, f2h_stm_hw_events[28], for FPGA to trigger events to STM.

FPGA-CTI

The FPGA-CTI allows the FPGA to send and receive triggers from the debug system.

Table 10-3: FPGA-CTI Signal Description Table

The following table lists the signal descriptions between the FPGA-CTI and FPGA.

Signal	Description
h2f_cti_trig_in[8]	Trigger input from FPGA
h2f_cti_trig_in_ack[8]	ACK signal to FPGA
h2f_cti_trig_out[8]	Trigger output to FPGA
h2f_cti_trig_out_ack[8]	ACK signal from FPGA
h2f_cti_clk	Clock input from FPGA
h2f_cti_fpga_clk_en	Clock enable driven by FPGA

Related Information

ARM Infocenter

For more information about the cross-trigger interface

CTI-NoC

The CTI is used for receiving triggers from device that can generate triggers and for sending triggers to devices that can receive triggers. This instance of CTI is used exclusively by NoC and allows NoC probe units to send and receive triggers.

Table 10-4: Trigger Input Interface

Name	Input/Output	Description
CTITRIGIN	Input	Trigger In Connected to noc_CTITRIGIN

Name	Input/Output	Description
TISBYPASSIN	Input	8'b0000_0000 —Trigger In synchronization bypass (Static value) Enable synchronizers for all 8 inputs. CTI runs off cs_at_clk
CTITRIGINACK	Output	Trigger In acknowledge Connected to noc_CTITRIGINACK

Table 10-5: Trigger Output Interface

Name	Input/Output	Description
CTITRIGOUT	Output	Trigger Out Connected to noc_CTITRIGOUT
CTITRIGOUTACK	Input	Trigger Out acknowledge Connected to noc_CTITRIGOUTACK
TISBYPASSACK	Input	8'b0000_0000—Trigger Out acknowledge synchronization bypass (Static value) Enable synchronizers for all 8 inputs
TIHSBYPASS	Input	8'b0000_0000—Trigger interface handshake bypass (Static value) Hand shaking is enabled

TPIU

Figure 10-6: Trace Clock and Trace Data Width

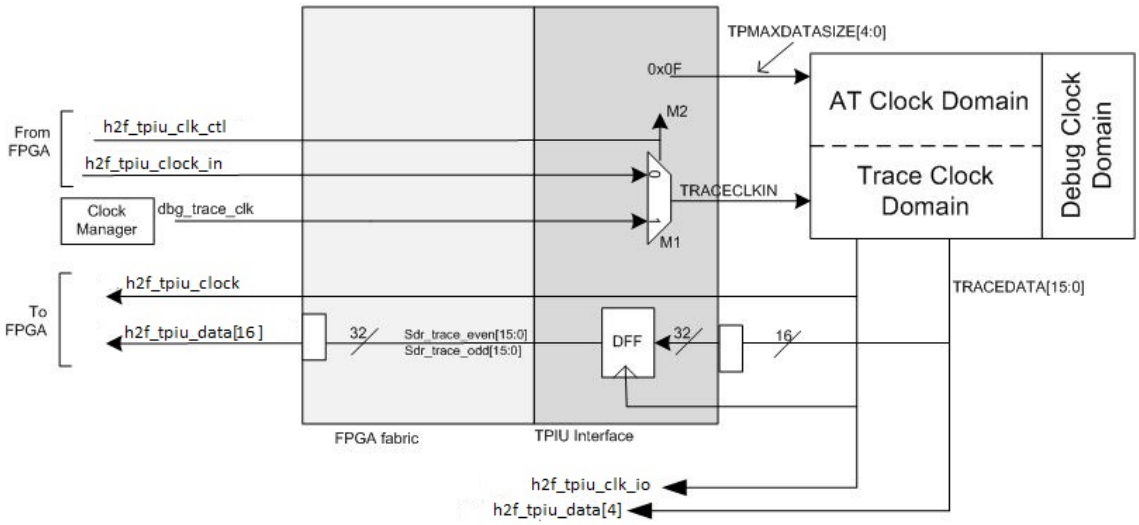


Table 10-6: TPIU Signals

The following table lists the signal descriptions between the TPIU and FPGA.

Signal	Description
h2f_tpiu_clk_ctl	Selects whether trace data is captured using the internal TPIU clock, which is the dbg_trace_clk signal from the clock manager; or an external clock provided as an input to the TPIU from the FPGA. 0 - use h2f_tpiu_clock_in 1 - use internal clock **Note**: When the FPGA is powered down or not configured the TPIU uses the internal clock.
h2f_tpiu_data[16]	16-bit trace data bus. Trace data changes on both edges of h2f_tpiu_clock.
h2f_tpiu_clock_in	Clock from the FPGA used to capture trace data.
h2f_tpiu_clock	Clock output from TPIU

Debug Clocks

Table 10-7: CoreSight Clocks

Port Name	Clock Source	Signal Name	Description
ATCLK	Clock manager	cs_at_clk	Trace bus clock.
CTICLK (for CTI)	Clock manager	cs_at_clk	Cross trigger interface clock for CTI. It can be synchronous or asynchronous to CTMCLK.
CTICLK (for FPGA-CTI)	FPGA fabric	cs_at_clk	Cross trigger interface clock for FPGA-CTI.
CTICLK (for CTI-0 and CTI-1)	Clock manager	mpu_clk	Cross trigger interface clock for CTI-0 and CTI-1. It can be synchronous or asynchronous to CTMCLK.
CTMCLK (for csCTM)	Clock manager	cs_pdbg_clk	Cross trigger matrix clock for csCTM. It can be synchronous or asynchronous to CTICLK.
CTMCLK (for CTM)	Clock manager	mpu_clk	Cross trigger matrix clock for CTM. It can be synchronous or asynchronous to CTICLK.
DAPCLK	Clock manager	cs_pdbg_clk	DAP internal clock. It must be equivalent to PCLKDBG.
PCLKDBG	Clock manager	cs_pdbg_clk	Debug APB (DAPB) clock.
HCLK	Clock manager	cs_pdbg_clk	Used by the AHB-Lite master inside the DAP. It is asynchronous to DAPCLK. In the HPS, the AHB-Lite port uses same clock as DAPCLK.
PCLKSYS	Clock manager	cs_pdbg_clk	Used by the APB slave port inside the DAP. It is asynchronous to DAPCLK.
SWCLKTCK	JTAG interface	dap_tck	The SWJ-DP clock driven by the external debugger through the JTAG interface. It is asynchronous to DAPCLK. When through the JTAG interface, this clock is the same as TCK of the JTAG interface.

Port Name	Clock Source	Signal Name	Description
TRACECLKIN	Clock manager	cs_trace_clk	TPIU trace clock input. It is asynchronous to ATCLK. In the HPS, this clock can come from the clock manager or the FPGA fabric.

Related Information

ARM Infocenter

For more information about the CoreSight port names, refer to the *CoreSight Technology System Design Guide*.

Debug Resets

The CoreSight system uses several resets.

Table 10-8: CoreSight Resets

ARM Reset Name	Clock Source	HPS Reset Signal Name	Description
ATRESETn	Reset manager	dbg_rst_n	Trace bus reset. It resets all registers in the ATCLK domain.
nCTIRESET	Reset manager	dbg_rst_n	CTI reset signal. It resets all registers in the CTICLK domain. In the HPS, there are four instances of CTI. All four use the same reset signal.
DAPRESETn	Reset manager	dbg_rst_n	DAP internal reset. It is connected to PRESETDBGn.
PRESETDBGn	Reset manager	dbg_rst_n	Debug APB reset. Resets all registers clocked by PCLKDBG.
HRESETn	Reset manager	sys_dbg_rst_n	SoC-provided reset signal that resets all of the AMBA on-chip interconnect. Use this signal to reset the DAP AHB-Lite master port.
PRESETSYSn	Reset manager	sys_dbg_rst_n	Resets system APB slave port of DAP.
nCTMRESET	Reset manager	dbg_rst_n	CTM reset signal. It resets all signals clocked by CTMCLK.

ARM Reset Name	Clock Source	HPS Reset Signal Name	Description
nPOTRST	Reset manager	tap_cold_rst_n	True power on reset signal to the DAP SWJ-DP. It must only reset at power-on.
nTRST	JTAG interface	nTRST pin	Resets the DAP TAP controller inside the SWJ-DP. This signal is driven by the host using the JTAG connector.
TRESETn	Reset manager	dbg_rst_n	Reset signal for TPIU. Resets all registers in the TRACECLKIN domain.

The ETR stall enable field (etrstallen) of the ctrl register in the reset manager controls whether the ETR is requested to stall its AXI master interface to the L3 interconnect before a warm or debug reset.

The level 4 (L4) watchdog timers can be paused during debugging to prevent reset while the processor is stopped at a breakpoint.

Related Information

- **Reset Manager** on page 3-1
- **Watchdog Timer** on page 24-1
- **ARM Infocenter**
  For more information about the CoreSight port names, refer to the *CoreSight Technology System Design Guide*.

CoreSight Debug and Trace Programming Model

This section describes programming model details specific to Altera's implementation of the ARM CoreSight technology.

The debug components can be configured to cause triggers when certain events occur. For example, soft logic in the FPGA fabric can signal an event which triggers an STM message injection into the trace stream.

Related Information

ARM Infocenter
Programming interface details of each CoreSight component.

Coresight Component Address

CoreSight components are configured through memory-mapped registers, located at offsets relative to the CoreSight component base address. CoreSight component base addresses are accessible through the component address table in the DAP ROM.

Table 10-9: Coresight Component Address Table

The following table is located in the ROM table portion of the DAP.

ROM Entry	Offset[30:12]	Description
0x0	0x00001	ETF Component Base Address
0x1	0x00002	CTI Component Base Address
0x2	0x00003	TPIU Component Base Address
0x3	0x00004	Trace Funnel Component Base Address
0x4	0x00005	STM Component Base Address
0x5	0x00006	ETR Component Base Address
0x6	0x00007	FPGA-CTI Component Base Address
0x7	0x00100	A9 ROM
0x8	0x00080	FPGA ROM
0x9	0x00008	L3-CTI

A host debugger can access this table at 0x80000000 through the DAP. HPS masters can access this ROM at 0xFF000000. Registers for a particular CoreSight component are accessed by adding the register offset to the CoreSight component base address, and adding that total to the base address of the ROM table.

The base address of the ROM table is different when accessed from the debugger (at 0x8000_0000) than when accessed from any HPS master (at 0xFF000000). For example, the CTI output enable (CTIOUTEN) register, `CTIOUTEN[2]` at offset 0xA8, can be accessed by the host debugger at 0x800020A8. To derive that value, add the host debugger access address to the ROM table of 0x80000000, to the CTI component base address of 0x00002000, to the `CTIOUTEN[2]` register offset of 0xA8.

STM Channels

The STM AXI slave is connected to the MPU, DMA, and FPGA-to-HPS bridge masters. Each master has up to 65536 channels where each channel occupies 256 bytes of address space, for a total of 16 MB per master. The HPS address map allocates 48 MB of consecutive address space to the STM AXI slave port, divided in three 16 MB segments.

Table 10-10: STM AXI Slave Port Address Allocation

Segment	Start Address	End Address
0	0xFC000000	0xFCFFFFFF
1	0xFD000000	0xFDFFFFFF

Segment	Start Address	End Address
2	0xFE000000	0xFEFFFFFF

Each of the three masters can access any one of the three address segments. Your software design determines which master uses which segment, based on the value of bits 24 and 25 in the write address, `AWADDRS[25:24]`. Software must restrict each master to use only one of the three segments.

Table 10-11: STM AXI Address Fields

STM receives trace data over an AXI slave port. This table contains a list of signals associated with this interface.

AXI Signal Fields	Description
`AWADDRS[7:0]`	These bits index the 256 bytes of the stimulus port.
`AWADDRS[23:8]`	These bits identify the 65536 stimulus ports associated with a master.
`AWADDRS[25:24]`	These bits identify the three masters. Only 0, 1, and 2 are valid values.
`AWADDRS[31:26]`	Always 0x3F. Bits 24 to 31 combine to access 0xFC000000 through 0xFEFFFFFF.

Each STM message contains a master ID that tells the host debugger which master is associated with the message. The STM master ID is determined by combining a portion of the `AWADDRS` signal and the `AWPROT` protection bit. In the following table, `MasterID[6]` identifies the

Table 10-12: STM Master ID Calculation

Master ID Bits	AXI Signal Bits	Notes
Master ID[5:0]	`AWADDRS[29:24]`	The lowest two bits are sufficient to determine which master, but CoreSight uses a six-bit master ID.
Master ID[6]	`AWPROT[1]`	0 indicates a secure master; 1 indicates a nonsecure master.

In addition to access through STM channels, the higher-order 28 (31:4) of the 32 event signals are attached to the FPGA through the FPGA-CTI. These event signals allow the FPGA fabric to send additional messages using the STM.

Related Information

- **HPS-FPGA Bridges** on page 8-1
- **ARM Infocenter**
  For more information, refer to "System Trace Macrocell" in the *Programmers' Model Architecture Specification* .

- **FPGA-CTI** on page 10-13

CTI Trigger Connections to Outside the Debug System

The following CTIs in the HPS debug system connect to outside the debug system:

- CTI
- FPGA-CTI
- L3-CTI

CTI

This section lists the trigger input, output, and output acknowledge pin connections implemented for CTI in the debug system. The trigger input acknowledge signals are not connected to pins.

Table 10-13: CTI Trigger Input Signals

The following table lists the trigger input pin connections implemented for CTI .

Number	Signal	Source
7	ASYNCOUT	STM
6	TRIGOUTHETE	STM
5	TRIGOUTSW	STM
4	TRIGOUTSPTE	STM
3	ACQCOMP	ETR
2	FULL	ETR
1	ACQCOMP	ETF
0	FULL	ETF

Table 10-14: CTI Trigger Output Signals

The following table lists the trigger output pin connections implemented for CTI .

Number	Signal	Destination
7	TRIGIN	ETF
6	FLUSHIN	ETF
5	HWEVENTS[3:2]	STM
4	HWEVENTS[1:0]	STM
3	TRIGIN	TPIU

Number	Signal	Destination
2	FLUSHIN	TPIU
1	TRIGIN	ETR
0	FLUSHIN	ETR

Table 10-15: CTI Trigger Output Acknowledge Signals

The following table lists the trigger output pin acknowledge connections implemented for CTI .

Number	Signal	Source
7	0	—
6	0	—
5	0	—
4	0	—
3	TRIGINACK	TPIU
2	FLUSHINACK	TPIU
1	0	—
0	0	—

FPGA-CTI

FPGA-CTI connects the debug system to the FPGA fabric. FPGA-CTI has all of its triggers available to the FPGA fabric.

Related Information

Configuring Embedded Cross-Trigger Connections on page 10-22
For more information about the triggers, refer to the "Configuring Embedded Cross-Trigger Connections" chapter.

L3-CTI

L3-CTI has all of its triggers available to the L3 interconnect.

Configuring Embedded Cross-Trigger Connections

CTI interfaces are programmable through a memory-mapped register interface.

The specific registers are described in the *CoreSight Components Technical Reference Manual*, which you can download from the ARM Infocenter.

To access registers in any CoreSight component through the debugger, the register offsets must be added to the CoreSight component's base address. That combined value must then be added to the address at which the ROM table is visible to the debugger (0x80000000).

Each CTI has two interfaces, the trigger interface and the channel interface. The trigger interface is the interface between the CTI and other components. It has eight trigger signals, which are hardwired to other components. The channel interface is the interface between a CTI and its CTM, with four bidirectional channels. The mapping of trigger interface to channel interface (and vice versa) in a CTI is dynamically configured. You can enable or disable each CTI trigger output and CTI trigger input connection individually.

For example, you can configure trigger input 0 in the FPGA-CTI to route to channel 3, and configure trigger output 3 in the FPGA-CTI and trigger output 7 in CTI-0 in the MPU debug subsystem to route from channel 3. This configuration causes a trigger at trigger input 0 in FPGA-CTI to propagate to trigger output 3 in the FPGA-CTI and trigger output 7 in CTI-0. Propagation can be single-to-single, single-to-multiple, multiple-to-single, and multiple-to-multiple.

A particular soft logic signal in the FPGA connected to a trigger input in the FPGA-CTI can be configured to trigger a flush of trace data to the TPIU. For example, you can configure channel 0 to trigger output 2 in the CTI. Then configure trigger input T3 to channel 0 in FPGA-CTI. Trace data is flushed to the TPIU when a trigger is received at trigger output 2 in the CTI.

Another soft logic signal in the FPGA connected to trigger input T2 in the FPGA-CTI can be configured to trigger a STM message. The CTI output triggers 4 and 5 are wired to the STM CoreSight component in the HPS. For example, configure channel 1 to trigger output 4 in the CTI; and then configure trigger input T2 to channel 1 in FPGA-CTI.

Another soft logic signal in the FPGA fabric connected to trigger input T1 in FPGA-CTI can be configured to trigger a breakpoint on CPU 1. Trigger output 1 in CTI-1 is wired to the debug request (EDBGRQ) signal of CPU-1. For example, configure channel 2 to trigger output 1 in CTI-1. Then configure trigger input T1 to channel 2 in FPGA-CTI.

Related Information

- **Coresight Component Address** on page 10-18
- **ARM Infocenter**
  For more information about the cross-trigger interface

Configuring Trigger Input 0

For example, you can configure trigger input 0 in the FPGA-CTI to route to channel 3, and configure trigger output 3 in the FPGA-CTI and trigger output 7 in CTI-0 in the MPU debug subsystem to route from channel 3. This configuration causes a trigger at trigger input 0 in FPGA-CTI to propagate to trigger output 3 in the FPGA-CTI and trigger output 7 in CTI-0. Propagation can be single-to-single, single-tomultiple, multiple-to-single, and multiple-to-multiple.

Triggering a Flush of Trace Data to the TPIU

A particular soft logic signal in the FPGA connected to a trigger input in the FPGA-CTI can be configured to trigger a flush of trace data to the TPIU. For example, you can configure channel 0 to trigger output 2 in CTI. Then configure trigger input T3 to channel 0 in FPGA-CTI. Trace data is flushed to the TPIU when a trigger is received at trigger output 2 in CTI.

Triggering an STM message

Another soft logic signal in the FPGA connected to trigger input T2 in FPGA-CTI can be configured to trigger an STM message. CTI output triggers 4 and 5 are wired to the STM CoreSight component in the HPS. For example, configure channel 1 to trigger output 4 in CTI. Then configure trigger input T2 to channel 1 in FPGA-CTI.

Triggering a Breakpoint on CPU 1

Another soft logic signal in the FPGA fabric connected to trigger input T1 in FPGA-CTI can be configured to trigger a breakpoint on CPU 1. Trigger output 1 in CTI-1 is wired to the external debug request (EDBGRQ) signal of CPU-1. For example, configure channel 2 to trigger output 1 in CTI-1. Then configure trigger input T1 to channel 2 in FPGA-CTI.

CoreSight Debug and Trace Address Map and Register Definitions

For complete HPS address map and register definitions, refer to the **Arria 10 HPS Address Map and Register Definitions** .

Document Revision History

Table 10-16: Document Revision History

Date	Version	Changes
October 2016	2016.10.28	Maintenance release
May 2016	2016.05.03	Maintenance release
November 2015	2015.11.02	Added a description on how the 2 TAP controllers are connected and supporting figures.
May 2015	2014.05.04	Maintenance release.
December 2014	2014.12.15	Maintenance release.
August 2014	2014.08.18	Initial release.

Error Checking and Correction Controller 11

2016.10.28

a10_5v4

Error Checking and Correction (ECC) controllers provide single- and double-bit error memory protection for integrated on-chip RAM and peripheral RAMs within the hard processor system (HPS).

ECC Controller Features

The features supported by each ECC controller are:

- Hamming code-based ECC calculations
- Single-bit error detection and correction
- Double-bit error detection
- Dedicated hardware block for memory data initialization
- Indirect memory access for:
  - Data correction on the corrupted memory address
  - Data and ECC syndrome bit error injection
- Watchdog timeout for indirect access to prevent bus stall
- Display of the current single or double-bit error memory address
- Single-bit error occurrence counter
- Look-up table (LUT) for logging single-bit error memory address
- Interrupt generated upon single and double-bit errors
- User-controllable interrupt assertion for test purposes

ECC Supported Memories

In addition to the 256 KB on-chip RAM, the peripherals that have integrated memories with ECC controllers are:

- USB OTG 0/1
- SD/MMC controller
- Ethernet MAC 0/1/2
- DMA controller
- NAND flash controller
- QSPI flash controller

Note: The L2 cache and the SDRAM interface have their own dedicated ECC support.

Related Information

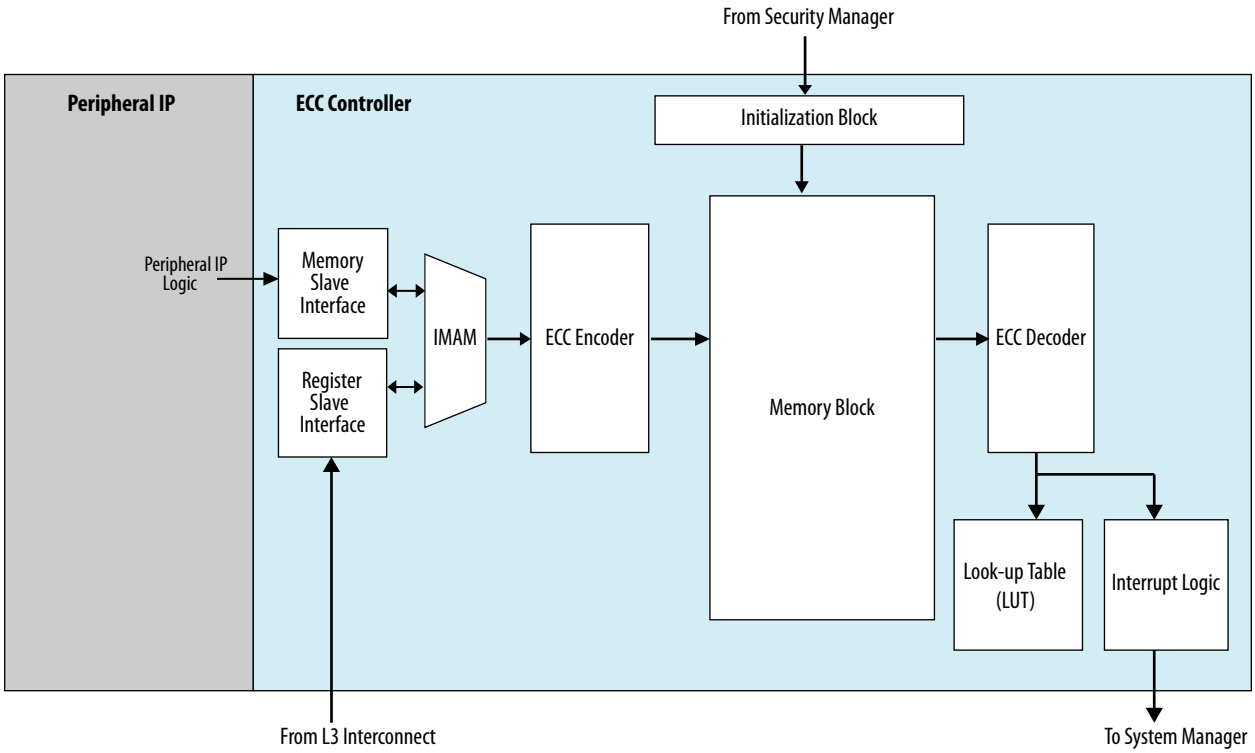
- **Cortex-A9 Microprocessor Unit Subsystem** on page 9-1
  For more information regarding L2 Cache ECC support.
- **Cortex-A9 Microprocessor Unit Subsystem** on page 9-1
  For more information regarding L2 Cache ECC support.
- **System Interconnect** on page 7-1
  For more information regarding SDRAM interface ECC support.

ECC Controller Block Diagram and System Integration

The figure below shows the ECC controller components and the ECC controller communication with other HPS peripherals.

Figure 11-1: ECC Block Diagram and System Integration

This figure applies to any of the peripheral IP that have ECC-supported memories.



Each peripheral accesses its memory block through the memory slave interface. The register slave interface allows the Microprocessor Unit (MPU) subsystem to access registers in the ECC controller for software configuration of the ECC controller. The register slave interface also allows the MPU subsystem to indirectly access the memory block through the indirect memory access MUX (IMAM).

Before the peripheral writes data to its memory block, it is encoded in the ECC controller. Before memory sends read data to a peripheral, it is decoded by the ECC controller. The initialization block initializes the memory data content, as well as the ECC syndrome bits, to known values. This block is controlled by the

register slave interface and also by the Security Manager when memory clearing is required during a tamper event.

When enabled, the look-up table (LUT) records the memory address of all single-bit errors, allowing you to analyze the error rate history.

The interrupt logic provides interrupt capability for single- and double-bit errors.

Related Information

Indirect Memory Access on page 11-6

ECC Controller Functional Description

Overview

An ECC controller can be enabled or disabled by programming the ECC Control (`CTRL`) register. The controller is disabled by default when the HPS is released from reset. When the ECC controller is disabled, data written to the memory block is not encoded, and data read from the memory block does not require ECC decoding. When the ECC controller is enabled, single-bit errors can be detected and corrected by the ECC controller. Double-bit errors are detected but not corrected.

ECC Structure

The ECC is calculated based on a Hamming code for the corresponding data word length.

Table 11-1: ECC Bits Required Based on Data Width

Data Bus Width	ECC Bits
8 to 15 bits	5
16 to 31 bits	6
32 to 63 bits	7
64 to 127 bits	8
128 to 255 bits	9
256 bits	10

Table 11-2: ECC Memory Characteristics

This table shows the memory data size and the Hamming code word length for each of the ECC-protected memories in the HPS, as well as the memory type. The Hamming code word length is calculated based on the full data width and whether the memory is byte- or word- addressable.

Notice that the on-chip RAM and DMA are byte-addressable. For each byte of data, five syndrome bits are used. For a data size of 64 bits (8 bytes), a total of `8 bytes*(8-bit data + 5-bit ECC)` bits are used for a Hamming code word.

Peripheral Memory	Data Size	Memory	ECC Bits	Data Width + ECC Bits	Hamming Code Word (length in bits)	Type [21]
On-chip RAM	64 x 32768	Byte-addressable	5 per byte lane	64+40[22]	104	Single port
USB RAM	35 x 8192	Word-addressable	7	35+7	42	Single port
SD/MMC FIFO	32 x 1024	Word-addressable	7	32+7	39	True dual port
EMAC Rx FIFO	35 x 4096	Word-addressable	7	35+7	42	Simple dual port
EMAC Tx FIFO	35 x 1024	Word-addressable	7	35+7	42	Simple dual port
DMA FIFO	64 x 512	Byte-addressable	5 per byte lane	64+40[22]	104	Simple dual port
NAND ECC Buffer	16 x 768	Word-addressable	6	16+6	22	Simple dual port
NAND Write FIFO	32 x 128	Word-addressable	7	32+7	39	Simple dual port
NAND Read FIFO	32 x 32	Word-addressable	7	32+7	39	Simple dual port
QSPI RAM	32 x 128	Word-addressable	7	32+7	39	Single port

RAM and ECC Memory Organization Example

The on-chip RAM and DMA have a memory organization that is byte-writeable, where every byte of data requires 5 bits of ECC.

The tables below shows the memory organization of the byte-writable memory with a 64-bit data size and 5 bits of ECC data.

[21] True dual-port memory has two writeable and two readable ports. Simple dual port memory has one write-only port and one read-only port.

[22] This is the same as 8 byte lanes with 5 ECC bits per lane.

Table 11-3: Organization of Byte-Writeable Memory with 64-bit Data Size

Address	RAM Bits							
	[63:56]	[55:48]	[47:40]	[39:32]	[31:24]	[23:16]	[15:8]	[7:0]
0x0	data[7]	data[6]	data[5]	data[4]	data[3]	data[2]	data[1]	data[0]
0x8	data[15]	data[14]	data[13]	data[12]	data[11]	data[10]	data[9]	data[8]

Table 11-4: Memory Organization of 5-Bit ECC Data

Address	ECC Memory Bits							
	[31:29]	[28:24]	[23:21]	[20:16]	[15:13]	[12:8]	[7:5]	[4:0]
0x0	0x0	ecc_data[3]	0x0	ecc_data[2]	0x0	ecc_data[1]	0x0	ecc_data[0]
0x4	0x0	ecc_data[7]	0x0	ecc_data[6]	0x0	ecc_data[5]	0x0	ecc_data[4]
0x8	0x0	ecc_data[11]	0x0	ecc_ data[10]	0x0	ecc_data[9]	0x0	ecc_data[8]
0xC	0x0	ecc_ data[15]	0x0	ecc_ data[14]	0x0	ecc_ data[13]	0x0	ecc_data[12]

Memory Data Initialization

When an ECC controller is enabled, the memory data must be written first before any data read occurs. If the memory is not written, the ECC syndrome bits are random, potentially causing false single- or double-bit errors when the memory data is read.

Every byte of data in the RAM is protected with ECC. This protection can lead to spurious ECC errors under the following conditions:

- When the MPU prefetches any uninitialized locations.
- When the MPU (or any master) reads from an uninitialized byte.

To prevent spurious ECC errors, software must use the memory initialization block in the ECC controller to initialize the entire memory data and ECC bits. The initialization block clears the memory data. Enabling initialization in the ECC Control (`CTRL`) register is independent of enabling the ECC.

Note: Peripherals with true dual-port memories, such as SD/MMC, must initialize both memories explicitly.

Software controls initialization through the ECC Control (`CTRL`) register. This process cannot be interrupted or stopped after it starts, therefore software must wait for the initialization complete (`INITCOMPLETE*`) bit to be set in the Initialization Status (`INITSTAT`) register. Memory accesses are allowed after the initialization process is complete.

The initialization block is accessed through the register slave interface. Additionally, the Security Manager can access the initialization block. If there is an unauthorized access to the memory or a tamper attempt, the Security Manager signals the Reset Manager to begin scrambling the on-chip memories through the initialization block. After scrambling, the initialization block clears the memory data to zero. This security feature is useful in applications that require secure boot, configuration and authentication.

The Security Manager can also be configured to scramble and clear memory on a cold or warm reset, or both a cold and warm reset. This feature is configured through a combination of user fuses and configuration registers within the Security Manager.

In the Security Manager, you can also select whether the memory is cleared in series or parallel. If time savings is required, memories can be initialized in parallel. If power integrity and savings are required, memories can be initialized in series.

Related Information

SoC Security on page 6-1
For more information regarding Security Manager functions, refer to the *SoC Security* chapter.

Indirect Memory Access

The register slave interface on an ECC controller allows software to access the memory block indirectly.

Through this interface, software can alter the memory data content and the stored ECC syndrome bits. By directly altering the data and syndrome bits, software can manually correct corrupted data, and run tests and diagnostics on the ECC controller.

Accesses to syndrome bits are not supported by the conventional memory access through the memory slave interface and instead, the ECC encoder handles them automatically.

Watchdog Timer

To prevent stalled indirect memory accesses, each ECC controller has a watchdog timer.

For example, if the clock to the memory block is stopped, the watchdog timer can assert an interrupt indicating that memory failed to respond within the expected interval. The watchdog timer is in a separate clock domain from the memory, enabling it to continue running independently of any problem with the memory clock.

The watchdog timer can be enabled or disabled in the ECC Watchdog Control (`ECC_wdcrtl`) register. The watchdog timeout is 2048 clock cycles of the clock domain that is connected to the ECC control slave port. The watchdog timeout interval is not software-programmable.

Data Correction

The data in the memory block can be overwritten through an indirect memory access. This feature is particularly useful when a double-bit error is detected on the data. The ECC controller provides the memory address of the current double-bit error. The data can be corrected by writing to the given memory address using an indirect memory access.

Error Injection

The ECC controller allows you to explicitly inject errors into the memory block for testing purposes. You can alter the memory data or ECC syndrome bits to manually introduce errors. If you change one bit of the memory data to the opposite value, a single-bit error is detected and corrected by the ECC controller. You can also alter ECC syndrome bits to test if the ECC controller triggers an error detection signal as expected.

Memory Testing

Testing methods can include both direct and indirect methods of memory access through the peripheral slave interface and the register slave interface. You can use both paths to test the syndrome bits and the

single-bit and double-bit error logic. However, you can only control ECC syndrome bits by injecting them through the register slave interface.

Peripheral Slave Interface Tests

Data and ECC overwrite bits in the `ECC_accctrl` register are provided to test the functionality of the peripheral interface to the ECC-protected RAM.

ECC-Enabled Test

The following sequence can be used to test if the ECC decoder works correctly.

1. Enable the ECC by setting the `ECC_EN` bit in the `CTRL` register.
2. Write data to any ECC-protected RAM memory location. This action generates an ECC value that can be read through the `ECC_Rdataecc0bus` and `ECC_Rdataecc1bus` registers.
3. Read back the memory data. The data read back is expected to match the data originally written (with or without a single-bit error) or generate a double-bit error. Refer to the "Error Logging" section for more details about identifying errors.

ECC-Disabled Test

This sequence can be used to test that the ECC decoder does not produce output when disabled.

1. Disable the ECC by clearing the `ECC_EN` bit in the `CTRL` register.
2. Write to any ECC-protected RAM memory location. No ECC value is expected to be generated and no interrupt or error logging occurs.
3. The ECC value can be read through the `ECC_Rdataecc0bus` and `ECC_Rdataecc1bus` registers to verify that the ECC values do not correspond to the read memory data.

ECC Disable/Enable Test

This sequence shows that memory data written when the ECC controller is disabled generates an error if the ECC controller is subsequently enabled and the same memory data location is read.

1. Disable the ECC by clearing the `ECC_EN` bit in the `CTRL` register.
2. Write to any ECC-protected RAM memory location. No ECC value is expected to be generated and no interrupt or error logging occurs.
3. Enable the ECC by setting the `ECC_EN` bit in the `CTRL` register.
4. Read the data from ECC-protected RAM memory location you wrote in step 2.
5. Expect an error to be generated because the ECC value corresponding to the memory data is not correct. Refer to the "Error Logging" section for more details about identifying errors.

Related Information

- **ECC Controller Address Map and Register Descriptions** on page 11-18
- **Error Logging** on page 11-11
  For more information about error logging.

Register Interface Tests

You can correct memory errors and test the memory register interface through registers in the ECC Controller.

The following registers can be used to test and correct memory:

- ECC_Addrbus: Holds the address of the memory and ECC data.
- ECC_RData3bus through ECC_RData0bus: Holds memory data from a read access.
- ECC_WData3bus through ECC_WData0bus: Holds the data to be written to memory.
- ECC_RDataecc1bus and ECC_RDataecc0bus: Holds the ECC data from a read access.
- ECC_WDataecc1bus and ECC_WDataecc0bus: Holds the ECC data to be written to memory.
- ECC_accctrl: Configures the access as a read or a write and enables memory and ECC data overwrites.
- ECC_startacc: Initiates the register interface access of memory data or ECC data.

Single-Bit Error Test

This sequence tests the single-bit error detection and correction in the ECC decoder.

1. Enable the ECC by setting the ECC_EN bit in the CTRL register.
2. Set the Data Override (DATAOVR) bit in the ECC_accctrl register.
3. Write data to an address location in memory using a normal memory write. Expect the correct ECC data to be generated.
4. Write a data value that has one bit altered in the ECC_WData3bus through ECC_WData0bus registers and write the address of the memory location in the ECC_Addrbus.
5. Configure the ECC_accctrl register to a write and set the ENBUSA bit of the ECC_startacc register to initiate the write. If the memory is dual-ported, an ENBUSB bit could optionally be enabled depending on the port access.
6. Read the same memory location using a normal memory read access. Expect a single-bit error to be logged but the data read to be correct. Refer to the "Error Logging" section for more details about identifying errors.

Double-Bit Error Test

This sequence tests the double-bit error detection in the ECC decoder.

1. Enable the ECC by setting the ECC_EN bit in the CTRL register.
2. Set the Data override (DATAOVR) bit in the ECC_accctrl register.
3. Write data to an address location in memory using a normal memory write. Expect the correct ECC data to be generated.
4. Write a data value that has two bits altered in the ECC_WData3bus through ECC_WData0bus registers and write the address of the memory location in the ECC_Addrbus.
5. Configure the ECC_accctrl register to a write and set the ENBUSA bit of the ECC_startacc register to initiate the write. If the memory is dual-ported, an ENBUSB bit could optionally be enabled depending on the port access.
6. Read the same memory location using a normal memory read access. Expect a double-bit error to be logged with no data correction. Refer to the "Error Logging" section for more details about identifying errors.

Related Information

- **ECC Controller Address Map and Register Descriptions** on page 11-18
- **Error Logging** on page 11-11
  For more information about error logging.

Error Checking and Correction Algorithm

The HPS error checking algorithm is based on an extended Hamming code, which is single-error correcting and double-error detecting (SECDED).

The computation can be understood by a given data (d) and a calculation of the check bits (c) through the equation:

$c = d \times H_d^T$

where H is the parity check matrix, $H = \{H_d, H_c\}$.

If the code word, designated by v and calculated by:

$v = \{d, c\}$

transmits to a noisy channel (for example in a RAM that is subjected to soft errors by cosmic rays) and becomes a contaminated code word, v', we can recover or discover the errors from its syndrome, s, by using the equation:

$s = v' \times H^T$

Errors are indicated when s does not equal 0. The syndrome shows the position of the error in the data.

The following examples show the parity check matrix for different data sizes.

Figure 11-2: 8-Bit Hamming Matrix

	d0	d1	d2	d3	d4	d5	d6	d7	c0	c1	c2	c3	c4
S0	0	0	0	1	1	1	1	1	1	-	-	-	-
S1	0	1	1	0	0	1	1	1	-	1	-	-	-
S2	1	0	1	1	1	0	0	1	-	-	1	-	-
S3	1	1	1	0	1	0	1	0	-	-	-	1	-
S4	1	1	0	1	0	1	0	0	-	-	-	-	1

Figure 11-3: 16-bit Hamming Matrix

	d0	d1	d2	d3	d4	d5	d6	d7	d8	d9	d10	d11	d12	d13	d14	d15	c0	c1	c2	c3	c4	c5
S0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	-	-	-	-	-
S1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	-	1	-	-	-	-
S2	0	1	1	1	0	0	0	1	0	1	1	1	0	0	0	1	-	-	1	-	-	-
S3	1	0	1	1	0	1	1	0	1	0	0	1	0	0	1	0	-	-	-	1	-	-
S4	1	1	0	1	1	0	1	0	1	0	1	0	0	1	0	0	-	-	-	-	1	-
S5	1	1	1	0	1	1	0	1	0	1	0	0	1	0	0	0	-	-	-	-	-	1

Figure 11-4: 32-bit Hamming Matrix

	d0	d1	d2	d3	d4	d5	d6	d7	d8	d9	d10	d11	d12	d13	d14	d15	d16	d17	d18	d19	d20	d21	d22	d23	d24	d25	d26	d27	d28	d29	d30	d31	c0	c1	c2	c3	c4	c5	c6
S0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	-	-	-	-	-	-
S1	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	1	1	1	1	1	-	1	-	-	-	-	-
S2	0	0	0	0	1	1	1	1	1	0	0	0	0	0	1	1	1	1	0	0	0	0	0	1	1	1	1	0	0	0	0	1	-	-	1	-	-	-	-
S3	0	1	1	1	0	0	0	1	1	0	0	1	1	1	0	0	0	1	0	0	1	1	1	0	0	0	1	0	0	0	1	0	-	-	-	1	-	-	-
S4	1	0	1	1	0	1	1	0	0	1	1	0	0	1	0	0	1	0	1	1	0	0	1	0	0	1	0	0	0	1	0	0	-	-	-	-	1	-	-
S5	1	1	0	1	1	0	1	0	1	0	1	0	1	0	0	1	0	0	0	1	0	1	0	0	1	0	0	0	1	0	0	0	-	-	-	-	-	1	-
s6	1	1	1	0	1	1	0	1	0	1	0	1	0	0	1	0	0	0	1	0	1	0	0	1	0	0	0	1	0	0	0	0	-	-	-	-	-	-	1

Figure 11-5: 35-bit Hamming Matrix

	d0	d1	d2	d3	d4	d5	d6	d7	d8	d9	d10	d11	d12	d13	d14	d15	d16	d17	d18	d19	d20	d21	d22	d23	d24
S0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1
S1	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0
S2	0	0	0	0	1	1	1	1	1	1	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0
S3	0	1	1	1	0	0	0	1	1	1	0	0	0	1	1	1	0	0	0	1	0	0	0	1	1
S4	1	0	1	1	0	1	1	0	0	1	0	1	1	0	0	1	0	0	1	0	0	1	1	0	0
S5	1	1	0	1	1	0	1	0	1	0	1	0	1	0	1	0	0	1	0	0	1	0	1	0	1
s6	1	1	1	0	1	1	0	1	0	0	1	1	0	1	0	0	1	0	0	0	1	1	0	1	0

	d25	d26	d27	d28	d29	d30	d31	d32	d33	d34	c0	c1	c2	c3	c4	c5	c6
S0	1	1	1	1	1	1	1	1	1	1	1	-	-	-	-	-	-
S1	0	0	0	0	0	1	1	1	1	1	-	1	-	-	-	-	-
S2	0	1	1	1	1	0	0	0	0	1	-	-	1	-	-	-	-
S3	1	0	0	0	1	0	0	0	1	0	-	-	-	1	-	-	-
S4	1	0	0	1	0	0	0	1	0	0	-	-	-	-	1	-	-
S5	0	0	1	0	0	0	1	0	0	0	-	-	-	-	-	1	-
s6	0	1	0	0	0	1	0	0	0	0	-	-	-	-	-	-	1

Figure 11-6: 136-bit Hamming Matrix

	d0	d1	d2	d3	d4	d5	d6	d7	d8	d9	d10	d11	d12	d13	d14	d15	d16	d17	d18	d19	d20	d21	d22	d23	d24
S0	1	1	1	0	1	1	0	1	0	0	1	1	0	1	0	0	1	0	0	0	1	1	0	1	0
S1	1	1	0	1	1	0	1	0	1	0	1	0	1	0	1	0	0	1	0	0	1	0	1	0	1
S2	1	0	1	1	0	1	1	0	0	1	0	1	1	0	0	1	0	0	1	0	0	1	1	0	0
S3	0	1	1	1	0	0	0	1	1	1	0	0	0	1	1	1	0	0	0	1	0	0	0	1	1
S4	0	0	0	0	1	1	1	1	1	1	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0
S5	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0
S6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1
S7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
S8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	d25	d26	d27	d28	d29	d30	d31	d32	d33	d34	d35	d36	d37	d38	d39	d40	d41	d42	d43	d44	d45	d46	d47	d48	d49
S0	0	1	0	0	0	1	0	0	0	0	1	1	0	1	0	0	1	0	0	0	1	0	0	0	0
S1	0	0	1	0	0	0	1	0	0	0	1	0	1	0	1	0	0	1	0	0	0	1	0	0	0
S2	1	0	0	1	0	0	0	1	0	0	0	1	1	0	0	1	0	0	1	0	0	0	0	0	0
S3	1	0	0	0	1	0	0	0	1	0	0	0	1	1	1	1	0	0	0	1	0	0	1	1	0
S4	0	1	1	1	1	0	0	0	0	1	0	0	0	0	0	0	1	1	1	1	0	0	0	0	1
S5	0	0	0	0	0	1	1	1	1	1	0	0	1	0	0	0	0	0	0	0	1	1	1	1	1
S6	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
S7	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1	1	1	1	1	1	1	1	1	1	1
S8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	d50	d51	d52	d53	d54	d55	d56	d57	d58	d59	d60	d61	d62	d63	d64	d65	d66	d67	d68	d69	d70	d71	d72	d73	d74
S0	1	0	0	0	0	0	1	1	0	1	0	0	1	0	0	0	1	0	0	0	0	1	0	0	0
S1	0	1	0	0	0	0	1	0	1	0	1	0	0	1	0	0	0	1	0	0	0	0	1	0	0
S2	0	0	1	0	0	0	0	1	1	0	0	1	0	0	1	0	0	0	1	0	0	0	0	1	0
S3	0	0	0	1	0	0	0	0	0	1	1	1	0	0	0	1	0	0	0	1	0	0	0	0	1
S4	0	0	0	0	1	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0	1	0	0	0	0
S5	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	0	0	0	0
S6	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1
S7	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
S8	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

	d75	d76	d77	d78	d79	d80	d81	d82	d83	d84	d85	d86	d87	d88	d89	d90	d91	d92	d93	d94	d95	d96	d97	d98	d99
S0	0	0	1	0	0	0	0	0	0	0	0	0	1	1	1	1	0	1	1	0	0	1	0	1	1
S1	0	0	0	1	0	0	0	0	0	0	1	1	0	0	0	1	0	0	1	0	1	1	1	1	1
S2	0	0	0	0	1	0	0	0	0	0	0	1	1	0	0	0	1	1	0	1	1	0	1	1	0
S3	0	0	0	0	0	1	0	0	0	1	1	1	1	1	0	1	0	1	1	1	1	0	0	1	1
S4	1	0	0	0	0	0	1	0	0	0	0	1	0	0	1	0	1	0	0	1	1	0	1	0	0
S5	0	1	0	0	0	0	0	1	0	0	0	1	1	1	1	0	0	0	1	0	0	1	0	0	1
S6	1	1	0	0	0	0	0	0	1	1	0	0	0	1	0	1	1	1	0	0	1	0	1	0	1
S7	0	0	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	0	1	1	0	1	0	0	0
S8	1	1	1	1	1	1	1	1	1	1	1	1	0	0	1	0	1	1	0	1	0	1	1	1	0

	d100	d101	d102	d103	d104	d105	d106	d107	d108	d109	d110	d111	d112	d113	d114	d115	d116	d117	d118	d119	d120	d121	d122	d123	d124
S0	1	0	1	1	1	0	0	0	0	1	1	0	0	1	0	0	1	1	0	1	1	0	0	0	1
S1	0	0	1	1	1	0	0	1	0	1	0	1	0	0	0	1	0	0	1	0	1	1	0	1	1
S2	0	1	1	0	1	1	0	0	1	1	1	1	1	1	0	1	0	1	0	0	1	0	1	0	0
S3	0	1	0	1	0	0	1	1	1	1	1	1	0	0	1	0	0	0	0	1	1	0	0	1	0
S4	1	0	0	1	0	1	1	0	1	1	0	0	1	0	1	1	1	0	1	1	0	0	0	1	0
S5	0	1	0	0	0	1	1	1	1	0	1	1	1	1	0	1	1	0	1	1	0	1	1	0	1
S6	1	1	1	0	1	1	1	0	1	0	0	1	1	0	1	1	1	1	0	0	0	1	1	0	1
S7	1	0	1	0	0	0	0	1	0	0	0	0	1	1	1	0	0	1	1	1	1	1	1	1	0
S8	1	1	0	1	1	1	1	1	0	0	1	0	0	1	1	0	1	1	1	0	0	1	1	1	1

	d125	d126	d127	d128	d129	d130	d131	d132	d133	d134	d135	c0	c1	c2	c3	c4	c5	c6	c7	c8
S0	0	0	1	1	1	1	1	0	1	1	1	1	-	-	-	-	-	-	-	-
S1	1	0	1	1	1	0	1	1	0	1	1	-	1	-	-	-	-	-	-	-
S2	1	1	0	0	1	1	1	0	1	0	0	-	-	1	-	-	-	-	-	-
S3	0	1	0	1	0	1	0	0	0	1	1	-	-	-	1	-	-	-	-	-
S4	1	1	1	1	1	1	1	1	1	0	1	-	-	-	-	1	-	-	-	-
S5	0	0	1	1	0	1	0	1	1	1	0	-	-	-	-	-	1	-	-	-
S6	1	1	1	0	0	0	1	1	0	0	0	-	-	-	-	-	-	1	-	-
S7	1	0	0	0	1	0	0	1	1	0	1	-	-	-	-	-	-	-	1	-
S8	0	1	0	0	0	0	0	0	0	1	0	-	-	-	-	-	-	-	-	1

Error Logging

The ECC controller logs the errors that are detected in the memory block. This logging is useful for diagnosing the error and possibly recovering the correct memory data.

Recent Error Address Registers

Each ECC controller logs the most recent single-bit and double-bit error memory addresses.

These address values are stored in the ECC Single-Bit Error Address (SERRADDRx) and Double-Bit Error Address (DERRADDRx) registers and can be read by software. These registers store the most recent memory

error address. The logged address is a word address, rather than a byte address value, so that software must shift the address left to obtain the true address corresponding to the address width. For example, a 16-bit address must be shifted left by one to represent the true address. For a 32-bit address the value must be shifted left by two; and for a 64-bit address, the value must be shifted left by three bits.

For a single-bit error, the SERRADDRx register logs the error address only if the single-bit error interrupt generation is enabled. Every double-bit error is logged if the ECC controller is enabled.

For true dual-port memory, two sets of recent error address registers are present. Each register shows the address of the error that has occurred on its corresponding memory port.

Related Information

ECC Structure on page 11-3
Refer to the "ECC Structure" section for more information regarding ECC Structure and Hamming Code word lengths.

Single-Bit Error Occurrence

The ECC controller has a 32-bit wide counter that increments on every occurrence of a single-bit error.

You can program the ECC controller to trigger an interrupt when the single-bit error counter has reached a specific value, which is configured in the Single-Bit Error Count (SERRCNTREG) register. You can reset the counter by clearing the CNT_RSTA bit in the ECC Control (CTRL) register.

For true dual-port memory, such as SD/MMC, two internal single-bit error counters are present in its ECC controller. Each counter counts the errors on its own memory port. However, both counters refer to the same user-configurable threshold for interrupt generation. In this case, program the counter threshold value in the SERRCNTREG register to represent the average number of errors of both memories.

Single-Bit Error Look-Up Table

The ECC controller of each memory port has a look-up table (LUT) that logs the memory addresses of all unique single-bit error occurrences. Repeated errors at the same memory address are not stored. The LUT keeps track of single-bit errors, but not double-bit errors.

The most significant bit (MSB) of each entry in the LUT is the valid bit. Whenever a single-bit error occurs and is logged by the LUT, the valid bit is set. The rest of the bits in a single LUT entry contain the memory address of the data error. After the memory address has been read from the LUT, software can clear the entry by writing a 1 to the valid bit in the entry. If all of the LUT entries are occupied and valid bits have not been cleared, overflow occurs on the next single-bit error. An interrupt can be generated on a LUT overflow. For more information about interrupts, refer to the "ECC Controller Interrupts" section.

The table below lists the LUT depth for every ECC-protected memory in the HPS.

Table 11-5: LUT Depth for HPS ECC Memories

Peripheral	LUT Depth (entries)
On-chip RAM	16
USB	8
SD/MMC	4 x 2 (23)
Ethernet MAC (Rx FIFO)	4

Peripheral	LUT Depth (entries)
Ethernet MAC (Tx FIFO)	4
DMA	4
NAND (ECC Buffer)	4
NAND (Write FIFO)	4
NAND (Read FIFO)	4
QSPI	4

The LUT entries are located in the ECC controller register map. Software can read the LUT error address and clear the valid bits. For more information, refer to the *ECC Controller Address Map and Register Description* section.

Related Information

- **ECC Controller Interrupts** on page 11-13
  For information about single- and double-bit error interrupts, refer to the "ECC Controller Interrupts" section.
- **ECC Controller Address Map and Register Descriptions** on page 11-18

ECC Controller Interrupts

The ECC controller has the ability to generate single- and double-bit error interrupts to the System Manager.

The ECC controller interrupt mechanism involves the System Manager, generic interrupt controller (GIC) and the ARM Cortex-A9 . The following steps outline the interrupt generation process when interrupts have been enabled through the Error Interrupt Enable (`ERRINTEN`) register.

1. The ECC controller generates an interrupt when an error occurs and notifies the System Manager.
2. The System Manager updates its interrupt status register and sends the interrupt to the GIC.
3. The GIC sends the interrupt to the MPU.
4. The MPU services the interrupt and clears the interrupt in the ECC controller.
5. The System Manager clears the interrupt to the GIC and the corresponding interrupt status bit.

Single-Bit Error Interrupts

The Single-Bit Error Interrupt Enable (`ERRINTEN`) register must be configured for single-bit error interrupt generation.

For true dual port memory, a separate interrupt is generated for errors on each memory port.

(23) The ECC controller for SD/MMC peripheral has two LUTs, 4-entries deep, because the memory used for the SD/MMC controller is a true dual-port type, where both ports can perform read operations. Reading either of the ports can trigger a single-bit error and so, a LUT is required for each of the ports.

The ECC controller can generate a single-bit error interrupt for:

- All single-bit errors
- LUT overflow
- Single-bit error counter match

The address of the most recent single-bit error is logged in the Single-Bit Error Address (SERRADDRx) register.

The interrrupt status (INSTAT) register indicates if a single-bit error is pending in the ECC controller. All single-bit interrupts are cleared by clearing the single-bit error pending bit of the INTSTAT register. The single-bit interrupt generation can be disabled by setting the error interrupt reset bit of the Error Interrupt Reset (ERRINTENR) register.

Related Information

ECC Controller Address Map and Register Descriptions on page 11-18

All Single-Bit Error Interrupt

To generate an interrupt for every single-bit error that occurs, regardless of whether it is with a new or repeated memory address access, you must:

- Clear the INTMODE bit in the Interrupt Mode (INTMODE) register
- Enable the interrupt by setting the SERRINTEN bit of the Error Interrupt Enable (ERRINTEN) register

This mode generates the most frequent interrupts and therefore, consumes greater processor cycle resources to service all the interrupts.

Note: Overflow data is not logged in this interrupt configuration.

LUT Overflow Interrupt

The LUT table can be used to generate two types of interrupts.

On every single-bit error detection and correction, the address of the error is logged in the LUT. Each address logged is unique and is at the data word boundary of its RAM bank. Coherency of the address table is maintained by a valid bit.

An interrupt can be generated for each new LUT entry or only when the LUT overflows. The following table describes the interrupt result based on the INTMODE and INTONOVF values in the INTMODE register. In this table, it is assumed that interrupts have been enabled by setting the SERRINTEN bit of the Error Interrupt Enable (ERRINTEN) register.

Note: If the INTMODE bit is clear, then all errors generate an interrupt and no overflow data is logged. The INTMODE bit must be set to 1 for the LUT to log entries.

Table 11-6: LUT Overflow Interrupt Configuration Options

INTMODE value	INTONOVF value	Result
0	X = Don't care	All errors generate an interrupt. No overflow data is logged.

INTMODE value	INTONOVF value	Result
1	0	An interrupt is generated for each new LUT entry. Overflow detection is disabled. Example: For a four-entry LUT, an interrupt asserts for each unique address entered in the LUT.
1	1	An interrupt is generated only when the LUT overflows. Example: If the LUT depth is four, the occurrence of the fifth unique address causes an interrupt to assert.

Counter Match Interrupt

The counter match interrupt allows you to set a threshold for the number of single-bit errors captured before an interrupt flag is set.

The INTONCMP bit in the INTMODE register enables the internal counter to count and compare against the SERRCNT value in the Single-Bit Error Count (SERRCNTREG) register. The internal counter increments on every single-bit error, regardless of whether it is a new or repeated address. The INTONCMP bit has no influence on the INTMODE and INTONOVF bits of the INTMODE register. As long as the internal counter is less than the Single-Bit Error Count (SERRCNTREG) register value, no interrupt is generated. When the internal counter is greater than or equal to the SERRCNTREG value, a single-bit interrupt request is asserted, the CMPFLGx bit is set in the Mode Status (MODSTAT) register, and the SERRPENx bit is set in the Interrupt Status (INTSTAT) register. When the match occurs, additional errors do not increment the counter until the CMPFLGx bit is cleared in the MODSTAT register.

This resultant match can be handled in three ways:

- Reset the error counter without restarting it. The ECC controller does not count single-bit errors until you restart the counter. Set the CNT_RSTx bit in the CTRL register to 1, which clears the counter. The CMPFLGx bit remains set. The counter does not increment until the CMPFLGx bit is cleared.
- Reset and restart the counter and clear the compare flag. Set CNT_RSTx bit in the CTRL register to 1, which clears the counter. Write a 1 to the CMPFLGx bit, which clears it. The internal counter begins counting from zero.
- Set the count to a higher value and clear the compare flag. Write the SERRCNTREG value to a higher value than the initial compare match value. Write a 1 to the CMPFLGx bit. This clears the CMPFLGx bit, but the internal counter is not reset and the count continues from where it left off until it reaches the new SERRCNTREG value.

If you allow the counter to resume during your interrupt service routine (ISR), it is possible that the error counter could run out again before the ISR exits. If this happens, and you clear the interrupt and exit the ISR, you will never detect the new counter match condition. To avoid this problem, check the CMPFLGx bit in the MODSTAT register prior to exiting the ISR. If CMPFLGx indicates another counter match condition, ensure that you handle it.

To clear the single-bit error interrupt, set the SERRPENx bit in the INTSTAT register.

Related Information

ECC Controller Address Map and Register Descriptions on page 11-18

Double-Bit Error Interrupt

All double-bit errors generate interrupts and the error memory address is logged into the recent double-bit error (DERRADDRx) register.

The Interrupt Status (INTSTAT) register indicates if a double-bit error has occured. The double-bit error interrupt generation cannot be disabled. The interrupt is de-asserted by writing to the double-bit error pending bit of the INTSTAT register.

Related Information

ECC Controller Address Map and Register Descriptions on page 11-18

Interrupt Testing

The ECC controller allows you to test the interrupt assertion and de-assertion to check if the interrupt logic is functioning properly. Set the single-or double-bit error test bits in the Interrupt Test (INTTEST) register to assert the corresponding interrupt. To clear the interrupt, set the single- or double-bit error pending bits in the Interrupt Status (INTSTAT) register.

ECC Controller Initialization and Configuration

The steps for initializing and configuring an ECC controller are as follows:

1. Turn off ECC interrupts by setting interrupt masks in the ecc_intmask_set register in the System Manager and disabling interrupts in the ERRINTEN register of the ECC Controller.
2. Ensure the ECC detection and correction logic is disabled by clearing the ECC_EN bit in the CTRL register.
3. Enable memory initialization through the ECC controller's memory initialization block by setting the INITx bit in the CTRL register. If the memory is dual-ported, initialization must be performed on both ports. Refer to the *ECC Structure* section to identify what type of memory you are initializing.
4. When the INITCOMPLETEx bit in the INITSTAT register is set, configure any single-bit, count, or compare match interrupts that are required. Enable ECC interrupts in the ECC controller and System Manager. Refer to the *ECC Controller Interrupts* section and the System Manager chapter for information on enabling interrupts.
5. (Optional) Test the ECC to see if it is functioning as expected. You can perform tests with or without the ECC controller enabled.

   Software can write to the control, data, and address registers provided in the ECC controller and then set the ENBUSx bit of the ECC_startacc register to initiate memory accesses. Alternatively, software can enable the ECC controller by setting the ECC_EN bit in the CTRL register and test the memory slave interface. Please refer to the *Memory Testing* section for more information on how to test the ECC controller.

   If optional testing is completed, clear the test memory.

After these steps are complete, normal accesses can occur.

When an ECC controller is enabled:

- The ECC controller writes the ECC bits whenever data is written to the RAM.
- Error interrupt requests can be enabled in the Interrupt Mode (INTMODE) register.
- Data errors are detected and correction is attempted.

The ECC calculation can only be performed when there is a valid RAM access.

Related Information

- **Memory Testing** on page 11-6

- **ECC Structure** on page 11-3
The ECC is calculated based on a Hamming code for the corresponding data word length.
- **ECC Controller Interrupts** on page 11-13
For details of single-bit, count, and compare-match interrupts.

ECC Controller Clocks

The ECC controller for each ECC-protected memory operates at the same clock frequency as its associated RAM port.

The ECC register interface, however, is in the `l4_mp_clk` domain. The clock source names for each ECC controller and its RAM are determined by the specific peripheral.

Table 11-7: Clock Source for Each ECC Controller and Memory

ECC Memory	Functional Clock
On-chip RAM	`l3_main_free_clk`
USB	asynchronous `l4_mp_clk`
SD/MMC	Port A read and write: `cclk_in`
	Port B read and write: `l4_mp_clk`
Ethernet MAC (Rx FIFO)	Read: `ap_clk`
	Write: `clk_rx_int`
Ethernet MAC (Tx FIFO)	Read: `ap_clk`
	Write: `clk_tx_int`
DMA	`l4_main_clk`
NAND (ECC Buffer)	`nand_clk`
NAND (Write FIFO)	Write: `nand_x_clk`
	Read: `nand_clk`
NAND (Read FIFO)	Read: `nand_x_clk`
	Write: `nand_clk`
QSPI	`l4_mp_clk`

Related Information

Clock Manager on page 2-1
For details of the clock signals for each ECC controller.

ECC Controller Reset

The Reset Manager drives the reset signal to the ECC controllers on a cold or warm reset. This resets the logic in the ECC controllers. The Security Manager can also request the Reset Manager to send a signal to scramble and clear all of the memories if a security event occurs.

Related Information

- **Reset Manager** on page 3-1
  For more information regarding reset, refer to the *Reset Manager* chapter.
- **SoC Security** on page 6-1
  For more information regarding Security Manager functions, refer to the *SoC Security* chapter.

ECC Controller Address Map and Register Descriptions

Each peripheral with ECC-supported memory has its own ECC Controller address space and registers. This section lists the individual peripheral's ECC Controller address maps and registers but this information can also be found within the peripheral chapters of this technical reference manual.

Related Information

- **Ethernet Media Access Controller** on page 17-1
  For more information regarding the EMAC controller.
- **NAND Flash Controller** on page 13-1
  For more information regarding the NAND Flash Controller.
- **SD/MMC Controller** on page 14-1
  For more information regarding the SD/MMC Controller.
- **On-Chip Memory** on page 12-1
  For more information regarding On-Chip RAM.
- **DMA Controller** on page 16-1
  For more information regarding the DMA Controller.
- **Quad SPI Flash Controller** on page 15-1
  For more information regarding the Quad SPI Flash Controller.
- **USB 2.0 OTG Controller** on page 18-1
  For more information regarding the USB 2.0 OTG Controller.

emac_rx_ecc Address Map

Module Instance	Base Address	End Address
ecc_emac0_rx_ecc_registerBlock	0xFF8C0800	0xFF8C0BFF
ecc_emac1_rx_ecc_registerBlock	0xFF8C1000	0xFF8C13FF
ecc_emac2_rx_ecc_registerBlock	0xFF8C1800	0xFF8C1BFF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
IP_REV_ID on page 11-38	0x0	32	RO	0x0	
CTRL on page 11-38	0x8	32	RW	0x0	ECC Control Register
INITSTAT on page 11-39	0xC	32	RW	0x0	This bit is used to set the initialize the memory and ecc to a known value
ERRINTEN on page 11-40	0x10	32	RW	0x0	Error Interrupt enable
ERRINTENS on page 11-41	0x14	32	RW	0x0	Error Interrupt set
ERRINTENR on page 11-42	0x18	32	RW	0x0	Error Interrupt reset.
INTMODE on page 11-43	0x1C	32	RW	0x0	Reads reflect SERRINTEN.
INTSTAT on page 11-43	0x20	32	RW	0x0	This bit is used to enable interrupt generation on SERR lookup table overflow. When all the entries in the table are valid=1 and this is bit is enabled, serr_req signal will be asserted.
INTTEST on page 11-44	0x24	32	RW	0x0	This bits is used to test interrupt from ECC RAM to GIC
MODSTAT on page 11-45	0x28	32	RW	0x0	Counter feature status flag
DERRADDRA on page 11-46	0x2C	32	RO	0x0	This register shows the address of PORTA current double-bit error. RAM size will determine the maximum number of address bits.

Register	Offset	Width	Access	Reset Value	Description
SERRADDRA on page 11-47	0x30	32	RO	0x0	This register shows the address of PORTA current single-bit error. RAM size will determine the maximum number of address bits.
SERRCNTREG on page 11-47	0x3C	32	RW	0x0	Maximum counter value for single-bit error interrupt
ECC_Addrbus on page 11-48	0x40	32	RW	0x0	MSB bit of address is determined by ADR.
ECC_RData0bus on page 11-49	0x44	32	RO	0x0	Data will be read to this register field.
ECC_RData1bus on page 11-50	0x48	32	RO	0x0	Data will be read to this register field.
ECC_RData2bus on page 11-50	0x4C	32	RO	0x0	Data will be read to this register field.
ECC_RData3bus on page 11-51	0x50	32	RO	0x0	Data will be read to this register field.
ECC_WData0bus on page 11-52	0x54	32	WO	0x0	Data from the register will be written to the RAM.
ECC_WData1bus on page 11-53	0x58	32	WO	0x0	Data from the register will be written to the RAM.
ECC_WData2bus on page 11-53	0x5C	32	WO	0x0	Data from the register will be written to the RAM.
ECC_WData3bus on page 11-54	0x60	32	WO	0x0	Data from the register will be written to the RAM.
ECC_RDataecc0bus on page 11-55	0x64	32	RO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Register	Offset	Width	Access	Reset Value	Description
ECC_RDataecc1bus on page 11-56	0x68	32	RO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.
ECC_WDataecc0bus on page 11-57	0x6C	32	WO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.
ECC_WDataecc1bus on page 11-58	0x70	32	WO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.
ECC_dbytectrl on page 11-59	0x74	32	RW	0x0	Max number of implemented byte enabled is DAT/8
ECC_accctrl on page 11-60	0x78	32	RW	0x0	These bits determine which byte of data/ecc to write to RAM.
ECC_startacc on page 11-61	0x7C	32	RW	0x0	These bits determine which byte of data/ecc to write to RAM.
ECC_wdctrl on page 11-61	0x80	32	RW	0x0	Bits to Enable/Disable Watch Dog Timer
SERRLKUPA0 on page 11-62	0x90	32	RW	0x0	Single-bit error address in LOOKUP TABLE for PORTA.

emac_rx_ecc Summary

Module Instance	Base Address
ecc_emac0_rx_ecc_registerBlock	0xFF8C0800
ecc_emac1_rx_ecc_registerBlock	0xFF8C1000
ecc_emac2_rx_ecc_registerBlock	0xFF8C1800

Register Address Offset	Bit Fields															
ecc_emac0_rx_ecc_register-Block																
IP_REV_ID 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SIREV RO 0x0															
CTRL 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															INITA 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							CNT_RSTA 0x0	Reserved							ECC_EN 0x0
INITSTAT 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															INITCOM-PLETEA 0x0
ERRINTEN 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTEN 0x0
ERRINTENS 0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTS 0x0

Register Address Offset	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ERRINTENR 0x18	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTR 0x0
INTMODE 0x1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															INTONCMP 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							INTONOVF 0x0	Reserved							INTMODE 0x0
INTSTAT 0x20	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							DERRPENA 0x0	Reserved							SERRPENA 0x0
INTTEST 0x24	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							TDERRA 0x0	Reserved							TSERRA 0x0
MODSTAT 0x28	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															CMPFLGA 0x0
DERRADDRA 0x2C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				Address 0x0											

Register Address Offset	Bit Fields															
SERRADDRA 0x30	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				Address 0x0											
SERRCNTREG 0x3C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	SERRCNT 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SERRCNT 0x0															
ECC_Addrbus 0x40	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				ECC_AddrBUS 0x0											
ECC_RData0bus 0x44	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_RData1bus 0x48	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													ECC_RDataBUS 0x0		
ECC_RData2bus 0x4C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															

Register Address Offset	Bit Fields															
ECC_RData3bus 0x50	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_WData0bus 0x54	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_WData1bus 0x58	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													ECC_WDataBUS 0x0		
ECC_WData2bus 0x5C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_WData3bus 0x60	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_RDataecc0bus 0x64	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_RDataecc3BUS 0x0							Reserved	ECC_RDataecc2BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_RDataecc1BUS 0x0							Reserved	ECC_RDataecc0BUS 0x0						

Register Address Offset	Bit Fields															
ECC_RDataecc1bus 0x68	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_RDataecc7BUS 0x0							Reserved	ECC_RDataecc6BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_RDataecc5BUS 0x0							Reserved	ECC_RDataecc4BUS 0x0						
ECC_WDataecc0bus 0x6C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_WDataecc3BUS 0x0							Reserved	ECC_WDataecc2BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_WDataecc1BUS 0x0							Reserved	ECC_WDataecc0BUS 0x0						
ECC_WDataecc1bus 0x70	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_WDataecc7BUS 0x0							Reserved	ECC_WDataecc6BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_WDataecc5BUS 0x0							Reserved	ECC_WDataecc4BUS 0x0						
ECC_dbytectrl 0x74	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															DBEN 0x0
ECC_accctrl 0x78	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							RDWR 0x0	Reserved						ECCOVR 0x0	DATAOVR 0x0
ECC_startacc 0x7C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															ENBUSA 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															

Register Address Offset	Bit Fields															
ECC_wdctrl 0x80	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															WDEN_RAM 0x0
SERRLKUPA0 0x90	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	VALID RW 0x0	Reserved														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				Address RO 0x0											
ecc_emac1_rx_ecc_register-Block																
IP_REV_ID 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SIREV RO 0x0															
CTRL 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															INITA 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							CNT_RSTA 0x0	Reserved							ECC_EN 0x0
INITSTAT 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															INITCOM-PLETEA 0x0

Register Address Offset	Bit Fields															
ERRINTEN 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTEN 0x0
ERRINTENS 0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTS 0x0
ERRINTENR 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTR 0x0
INTMODE 0x1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															INTONCMP 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							INTONOVF 0x0	Reserved							INTMODE 0x0
INTSTAT 0x20	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							DERRPENA 0x0	Reserved							SERRPENA 0x0
INTTEST 0x24	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							TDERRA 0x0	Reserved							TSERRA 0x0

Register Address Offset	Bit Fields															
MODSTAT 0x28	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															CMPFLGA 0x0
DERRADDRA 0x2C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				Address 0x0											
SERRADDRA 0x30	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				Address 0x0											
SERRCNTREG 0x3C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	SERRCNT 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SERRCNT 0x0															
ECC_Addrbus 0x40	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				ECC_AddrBUS 0x0											
ECC_RData0bus 0x44	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															

Register Address Offset	Bit Fields															
ECC_RData1bus 0x48	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													ECC_RDataBUS 0x0		
ECC_RData2bus 0x4C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_RData3bus 0x50	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_WData0bus 0x54	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_WData1bus 0x58	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													ECC_WDataBUS 0x0		
ECC_WData2bus 0x5C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															

Register Address Offset	Bit Fields															
ECC_WData3bus 0x60	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_RDataecc0bus 0x64	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_RDataecc3BUS 0x0							Reserved	ECC_RDataecc2BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_RDataecc1BUS 0x0							Reserved	ECC_RDataecc0BUS 0x0						
ECC_RDataecc1bus 0x68	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_RDataecc7BUS 0x0							Reserved	ECC_RDataecc6BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_RDataecc5BUS 0x0							Reserved	ECC_RDataecc4BUS 0x0						
ECC_WDataecc0bus 0x6C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_WDataecc3BUS 0x0							Reserved	ECC_WDataecc2BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_WDataecc1BUS 0x0							Reserved	ECC_WDataecc0BUS 0x0						
ECC_WDataecc1bus 0x70	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_WDataecc7BUS 0x0							Reserved	ECC_WDataecc6BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_WDataecc5BUS 0x0							Reserved	ECC_WDataecc4BUS 0x0						
ECC_dbytectrl 0x74	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															DBEN 0x0

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
ECC_accctrl 0x78	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							RDWR 0x0	Reserved						ECCOVR 0x0	DATAOVR 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_startacc 0x7C	Reserved															ENBUSA 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_wdctrl 0x80	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															WDEN_RAM 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SERRLKUPA0 0x90	VALID RW 0x0	Reserved														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				Address RO 0x0											
ecc_emac2_rx_ecc_register-Block																
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IP_REV_ID 0x0	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SIREV RO 0x0															

Register Address Offset	Bit Fields															
CTRL 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															INITA 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							CNT_RSTA 0x0	Reserved							ECC_EN 0x0
INITSTAT 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															INITCOM-PLETEA 0x0
ERRINTEN 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTEN 0x0
ERRINTENS 0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTS 0x0
ERRINTENR 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTR 0x0
INTMODE 0x1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															INTONCMP 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							INTONOVF 0x0	Reserved							INTMODE 0x0

Register Address Offset	Bit Fields															
INTSTAT 0x20	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							DERRPENA 0x0	Reserved							SERRPENA 0x0
INTTEST 0x24	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							TDERRA 0x0	Reserved							TSERRA 0x0
MODSTAT 0x28	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															CMPFLGA 0x0
DERRADDRA 0x2C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				Address 0x0											
SERRADDRA 0x30	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				Address 0x0											
SERRCNTREG 0x3C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	SERRCNT 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SERRCNT 0x0															

Register Address Offset	Bit Fields															
ECC_Addrbus 0x40	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				ECC_AddrBUS 0x0											
ECC_RData0bus 0x44	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_RData1bus 0x48	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													ECC_RDataBUS 0x0		
ECC_RData2bus 0x4C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_RData3bus 0x50	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_WData0bus 0x54	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															

Register Address Offset	Bit Fields															
ECC_WData1bus 0x58	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													ECC_WDataBUS 0x0		
ECC_WData2bus 0x5C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_WData3bus 0x60	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_RDataecc0bus 0x64	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_RDataecc3BUS 0x0							Reserved	ECC_RDataecc2BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_RDataecc1BUS 0x0							Reserved	ECC_RDataecc0BUS 0x0						
ECC_RDataecc1bus 0x68	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_RDataecc7BUS 0x0							Reserved	ECC_RDataecc6BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_RDataecc5BUS 0x0							Reserved	ECC_RDataecc4BUS 0x0						
ECC_WDataecc0bus 0x6C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_WDataecc3BUS 0x0							Reserved	ECC_WDataecc2BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_WDataecc1BUS 0x0							Reserved	ECC_WDataecc0BUS 0x0						

Register Address Offset	Bit Fields															
ECC_WDataecc1bus 0x70	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_WDataecc7BUS 0x0							Reserved	ECC_WDataecc6BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_WDataecc5BUS 0x0							Reserved	ECC_WDataecc4BUS 0x0						
ECC_dbytectrl 0x74	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															DBEN 0x0
ECC_accctrl 0x78	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							RDWR 0x0	Reserved						ECCOVR 0x0	DATAOVR 0x0
ECC_startacc 0x7C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															ENBUSA 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															
ECC_wdctrl 0x80	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															WDEN_RAM 0x0
SERRLKUPA0 0x90	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	VALID RW 0x0	Reserved														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				Address RO 0x0											

IP_REV_ID

Module Instance	Base Address	Register Address
ecc_emac0_rx_ecc_registerBlock	0xFF8C0800	0xFF8C0800
ecc_emac1_rx_ecc_registerBlock	0xFF8C1000	0xFF8C1000
ecc_emac2_rx_ecc_registerBlock	0xFF8C1800	0xFF8C1800

Offset: 0x0

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SIREV RO 0x0															

IP_REV_ID Fields

Bit	Name	Description	Access	Reset
15:0	SIREV	IP Rev # These bits indicate the silicon revision number.	RO	0x0

CTRL

ECC Control Register

Module Instance	Base Address	Register Address
ecc_emac0_rx_ecc_registerBlock	0xFF8C0800	0xFF8C0808
ecc_emac1_rx_ecc_registerBlock	0xFF8C1000	0xFF8C1008
ecc_emac2_rx_ecc_registerBlock	0xFF8C1800	0xFF8C1808

Offset: 0x8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															INITA 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							CNT_RSTA 0x0	Reserved							ECC_EN 0x0

CTRL Fields

Bit	Name	Description	Access	Reset
16	INITA	Enable for the hardware memory initialization PORTA.	RW	0x0
8	CNT_RSTA	Enable to reset internal single-bit error counter A value to zero	RW	0x0
0	ECC_EN	Enable for the ECC detection and correction logic.	RW	0x0

INITSTAT

This bit is used to set the initialize the memory and ecc to a known value

Module Instance	Base Address	Register Address
ecc_emac0_rx_ecc_registerBlock	0xFF8C0800	0xFF8C080C
ecc_emac1_rx_ecc_registerBlock	0xFF8C1000	0xFF8C100C
ecc_emac2_rx_ecc_registerBlock	0xFF8C1800	0xFF8C180C

Offset: 0xC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															INITCOM-PLETEA 0x0

INITSTAT Fields

Bit	Name	Description	Access	Reset
0	INITCOMPLETEA	This bit is used to verify if the hardware memory initialization has completed PORTB.	RW	0x0

ERRINTEN

Error Interrupt enable

Module Instance	Base Address	Register Address
ecc_emac0_rx_ecc_registerBlock	0xFF8C0800	0xFF8C0810
ecc_emac1_rx_ecc_registerBlock	0xFF8C1000	0xFF8C1010
ecc_emac2_rx_ecc_registerBlock	0xFF8C1800	0xFF8C1810

Offset: 0x10

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															SERRINTEN 0x0

ERRINTEN Fields

Bit	Name	Description	Access	Reset
0	SERRINTEN	This bit is used to enable the single bit error interrupt of ECC RAM system	RW	0x0

ERRINTENS

Error Interrupt set

Module Instance	Base Address	Register Address
ecc_emac0_rx_ecc_registerBlock	0xFF8C0800	0xFF8C0814
ecc_emac1_rx_ecc_registerBlock	0xFF8C1000	0xFF8C1014
ecc_emac2_rx_ecc_registerBlock	0xFF8C1800	0xFF8C1814

Offset: 0x14

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															SERRINTS 0x0

ERRINTENS Fields

Bit	Name	Description	Access	Reset
0	SERRINTS	This bit is used to set the single-bit error interrupt bit.	RW	0x0

ERRINTENR

Error Interrupt reset.

Module Instance	Base Address	Register Address
ecc_emac0_rx_ecc_registerBlock	0xFF8C0800	0xFF8C0818
ecc_emac1_rx_ecc_registerBlock	0xFF8C1000	0xFF8C1018
ecc_emac2_rx_ecc_registerBlock	0xFF8C1800	0xFF8C1818

Offset: 0x18

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															SERRINTR 0x0

ERRINTENR Fields

Bit	Name	Description	Access	Reset
0	SERRINTR	This bit is used to reset the single-bit error interrupt bit. Reads reflect SERRINTEN. 1'b0: Writing of zero has no effect. 1'b1: By writing one, this bit will reset SERRINTEN bit to 0. This is performing a bitwise writing of this feature.	RW	0x0

INTMODE

Reads reflect SERRINTEN.

Module Instance	Base Address	Register Address
ecc_emac0_rx_ecc_registerBlock	0xFF8C0800	0xFF8C081C
ecc_emac1_rx_ecc_registerBlock	0xFF8C1000	0xFF8C101C
ecc_emac2_rx_ecc_registerBlock	0xFF8C1800	0xFF8C181C

Offset: 0x1C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															INTONCMP 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							INTONOVF 0x0	Reserved							INTMODE 0x0

INTMODE Fields

Bit	Name	Description	Access	Reset
16	INTONCMP	Enable interrupt on compare.	RW	0x0
8	INTONOVF	Enable interrupt on overflow.	RW	0x0
0	INTMODE	Interrupt mode for single-bit errors.	RW	0x0

INTSTAT

This bit is used to enable interrupt generation on SERR lookup table overflow. When all the entries in the table are valid=1 and this is bit is enabled, serr_req signal will be asserted.

Module Instance	Base Address	Register Address
ecc_emac0_rx_ecc_registerBlock	0xFF8C0800	0xFF8C0820
ecc_emac1_rx_ecc_registerBlock	0xFF8C1000	0xFF8C1020
ecc_emac2_rx_ecc_registerBlock	0xFF8C1800	0xFF8C1820

Offset: 0x20

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							DERRPENA 0x0	Reserved							SERRPENA 0x0

INTSTAT Fields

Bit	Name	Description	Access	Reset
8	DERRPENA	Double-bit error pending for PORTA.	RW	0x0
0	SERRPENA	Single-bit error pending for PORTA.	RW	0x0

INTTEST

This bits is used to test interrupt from ECC RAM to GIC

Module Instance	Base Address	Register Address
ecc_emac0_rx_ecc_registerBlock	0xFF8C0800	0xFF8C0824
ecc_emac1_rx_ecc_registerBlock	0xFF8C1000	0xFF8C1024
ecc_emac2_rx_ecc_registerBlock	0xFF8C1800	0xFF8C1824

Offset: 0x24

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							TDERRA 0x0	Reserved							TSERRA 0x0

INTTEST Fields

Bit	Name	Description	Access	Reset
8	TDERRA	Test PORTA Double-bit error.	RW	0x0
0	TSERRA	Test PORTA Single-bit error.	RW	0x0

MODSTAT

Counter feature status flag

Module Instance	Base Address	Register Address
ecc_emac0_rx_ecc_registerBlock	0xFF8C0800	0xFF8C0828
ecc_emac1_rx_ecc_registerBlock	0xFF8C1000	0xFF8C1028
ecc_emac2_rx_ecc_registerBlock	0xFF8C1800	0xFF8C1828

Offset: 0x28

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															CMPFLGA 0x0

MODSTAT Fields

Bit	Name	Description	Access	Reset
0	CMPFLGA	Port A compare status flag	RW	0x0

DERRADDRA

This register shows the address of PORTA current double-bit error. RAM size will determine the maximum number of address bits.

Module Instance	Base Address	Register Address
ecc_emac0_rx_ecc_registerBlock	0xFF8C0800	0xFF8C082C
ecc_emac1_rx_ecc_registerBlock	0xFF8C1000	0xFF8C102C
ecc_emac2_rx_ecc_registerBlock	0xFF8C1800	0xFF8C182C

Offset: 0x2C

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				Address 0x0											

DERRADDRA Fields

Bit	Name	Description	Access	Reset
11:0	Address	Recent double-bit error address.	RO	0x0

SERRADDRA

This register shows the address of PORTA current single-bit error. RAM size will determine the maximum number of address bits.

Module Instance	Base Address	Register Address
ecc_emac0_rx_ecc_registerBlock	0xFF8C0800	0xFF8C0830
ecc_emac1_rx_ecc_registerBlock	0xFF8C1000	0xFF8C1030
ecc_emac2_rx_ecc_registerBlock	0xFF8C1800	0xFF8C1830

Offset: 0x30

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				Address 0x0											

SERRADDRA Fields

Bit	Name	Description	Access	Reset
11:0	Address	Recent single-bit error address.	RO	0x0

SERRCNTREG

Maximum counter value for single-bit error interrupt

Module Instance	Base Address	Register Address
ecc_emac0_rx_ecc_registerBlock	0xFF8C0800	0xFF8C083C
ecc_emac1_rx_ecc_registerBlock	0xFF8C1000	0xFF8C103C
ecc_emac2_rx_ecc_registerBlock	0xFF8C1800	0xFF8C183C

Offset: 0x3C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SERRCNT 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SERRCNT 0x0															

SERRCNTREG Fields

Bit	Name	Description	Access	Reset
31:0	SERRCNT	Counter value	RW	0x0

ECC_Addrbus

MSB bit of address is determined by ADR.

Module Instance	Base Address	Register Address
ecc_emac0_rx_ecc_registerBlock	0xFF8C0800	0xFF8C0840
ecc_emac1_rx_ecc_registerBlock	0xFF8C1000	0xFF8C1040
ecc_emac2_rx_ecc_registerBlock	0xFF8C1800	0xFF8C1840

Offset: 0x40

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				ECC_AddrBUS 0x0											

ECC_Addrbus Fields

Bit	Name	Description	Access	Reset
11:0	ECC_AddrBUS	Address will be driven to RAM to either read or write the data. Address will be latched by the RAM when the Enbus is asserted.	RW	0x0

ECC_RData0bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_emac0_rx_ecc_registerBlock	0xFF8C0800	0xFF8C0844
ecc_emac1_rx_ecc_registerBlock	0xFF8C1000	0xFF8C1044
ecc_emac2_rx_ecc_registerBlock	0xFF8C1800	0xFF8C1844

Offset: 0x44

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData0bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_RDataBUS	ECC_RDataBUS[31:0].	RO	0x0

ECC_RData1bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_emac0_rx_ecc_registerBlock	0xFF8C0800	0xFF8C0848
ecc_emac1_rx_ecc_registerBlock	0xFF8C1000	0xFF8C1048
ecc_emac2_rx_ecc_registerBlock	0xFF8C1800	0xFF8C1848

Offset: 0x48

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													ECC_RDataBUS 0x0		

ECC_RData1bus Fields

Bit	Name	Description	Access	Reset
2:0	ECC_RDataBUS	ECC_RDataBUS[63:32].	RO	0x0

ECC_RData2bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_emac0_rx_ecc_registerBlock	0xFF8C0800	0xFF8C084C
ecc_emac1_rx_ecc_registerBlock	0xFF8C1000	0xFF8C104C
ecc_emac2_rx_ecc_registerBlock	0xFF8C1800	0xFF8C184C

Offset: 0x4C

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData2bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_RDataBUS	ECC_RDataBUS[95:64].	RO	0x0

ECC_RData3bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_emac0_rx_ecc_registerBlock	0xFF8C0800	0xFF8C0850
ecc_emac1_rx_ecc_registerBlock	0xFF8C1000	0xFF8C1050
ecc_emac2_rx_ecc_registerBlock	0xFF8C1800	0xFF8C1850

Offset: 0x50

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData3bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_RDataBUS	ECC_RDataBUS[127-96].	RO	0x0

ECC_WData0bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_emac0_rx_ecc_registerBlock	0xFF8C0800	0xFF8C0854
ecc_emac1_rx_ecc_registerBlock	0xFF8C1000	0xFF8C1054
ecc_emac2_rx_ecc_registerBlock	0xFF8C1800	0xFF8C1854

Offset: 0x54

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData0bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_WDataBUS	ECC_WDataBUS[31:0].	WO	0x0

ECC_WData1bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_emac0_rx_ecc_registerBlock	0xFF8C0800	0xFF8C0858
ecc_emac1_rx_ecc_registerBlock	0xFF8C1000	0xFF8C1058
ecc_emac2_rx_ecc_registerBlock	0xFF8C1800	0xFF8C1858

Offset: 0x58

Access: WO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													ECC_WDataBUS 0x0		

ECC_WData1bus Fields

Bit	Name	Description	Access	Reset
2:0	ECC_WDataBUS	ECC_WDataBUS[63:32].	WO	0x0

ECC_WData2bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_emac0_rx_ecc_registerBlock	0xFF8C0800	0xFF8C085C
ecc_emac1_rx_ecc_registerBlock	0xFF8C1000	0xFF8C105C
ecc_emac2_rx_ecc_registerBlock	0xFF8C1800	0xFF8C185C

Offset: 0x5C

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData2bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_WDataBUS	ECC_WDataBUS[95-64].	WO	0x0

ECC_WData3bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_emac0_rx_ecc_registerBlock	0xFF8C0800	0xFF8C0860
ecc_emac1_rx_ecc_registerBlock	0xFF8C1000	0xFF8C1060
ecc_emac2_rx_ecc_registerBlock	0xFF8C1800	0xFF8C1860

Offset: 0x60

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData3bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_WDataBUS	ECC_WDataBUS[127-96].	WO	0x0

ECC_RDataecc0bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_emac0_rx_ecc_registerBlock	0xFF8C0800	0xFF8C0864
ecc_emac1_rx_ecc_registerBlock	0xFF8C1000	0xFF8C1064
ecc_emac2_rx_ecc_registerBlock	0xFF8C1800	0xFF8C1864

Offset: 0x64

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	ECC_RDataecc3BUS 0x0							Reserved	ECC_RDataecc2BUS 0x0						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	ECC_RDataecc1BUS 0x0							Reserved	ECC_RDataecc0BUS 0x0						

ECC_RDataecc0bus Fields

Bit	Name	Description	Access	Reset
30:24	ECC_RDataecc3BUS	Eccdata will be read to this register field.	RO	0x0
22:16	ECC_RDataecc2BUS	Eccdata will be read to this register field.	RO	0x0
14:8	ECC_RDataecc1BUS	Eccdata will be read to this register field.	RO	0x0
6:0	ECC_RDataecc0BUS	Eccdata will be read to this register field.	RO	0x0

ECC_RDataecc1bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_emac0_rx_ecc_registerBlock	0xFF8C0800	0xFF8C0868
ecc_emac1_rx_ecc_registerBlock	0xFF8C1000	0xFF8C1068
ecc_emac2_rx_ecc_registerBlock	0xFF8C1800	0xFF8C1868

Offset: 0x68

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	ECC_RDataecc7BUS 0x0							Reserved	ECC_RDataecc6BUS 0x0						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	ECC_RDataecc5BUS 0x0							Reserved	ECC_RDataecc4BUS 0x0						

ECC_RDataecc1bus Fields

Bit	Name	Description	Access	Reset
30:24	ECC_RDataecc7BUS	Eccdata will be read to this register field.	RO	0x0
22:16	ECC_RDataecc6BUS	Eccdata will be read to this register field.	RO	0x0
14:8	ECC_RDataecc5BUS	Eccdata will be read to this register field.	RO	0x0
6:0	ECC_RDataecc4BUS	Eccdata will be read to this register field.	RO	0x0

ECC_WDataecc0bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_emac0_rx_ecc_registerBlock	0xFF8C0800	0xFF8C086C
ecc_emac1_rx_ecc_registerBlock	0xFF8C1000	0xFF8C106C
ecc_emac2_rx_ecc_registerBlock	0xFF8C1800	0xFF8C186C

Offset: 0x6C

Access: WO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	ECC_WDataecc3BUS 0x0							Reserved	ECC_WDataecc2BUS 0x0						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	ECC_WDataecc1BUS 0x0							Reserved	ECC_WDataecc0BUS 0x0						

ECC_WDataecc0bus Fields

Bit	Name	Description	Access	Reset
30:24	ECC_WDataecc3BUS	Eccdata from the register will be written to the RAM.	WO	0x0
22:16	ECC_WDataecc2BUS	Eccdata from the register will be written to the RAM.	WO	0x0
14:8	ECC_WDataecc1BUS	Eccdata from the register will be written to the RAM.	WO	0x0
6:0	ECC_WDataecc0BUS	Eccdata from the register will be written to the RAM.	WO	0x0

ECC_WDataecc1bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_emac0_rx_ecc_registerBlock	0xFF8C0800	0xFF8C0870
ecc_emac1_rx_ecc_registerBlock	0xFF8C1000	0xFF8C1070
ecc_emac2_rx_ecc_registerBlock	0xFF8C1800	0xFF8C1870

Offset: 0x70

Access: WO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	ECC_WDataecc7BUS 0x0							Reserved	ECC_WDataecc6BUS 0x0						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	ECC_WDataecc5BUS 0x0							Reserved	ECC_WDataecc4BUS 0x0						

ECC_WDataecc1bus Fields

Bit	Name	Description	Access	Reset
30:24	ECC_WDataecc7BUS	Eccdata from the register will be written to the RAM.	WO	0x0
22:16	ECC_WDataecc6BUS	Eccdata from the register will be written to the RAM.	WO	0x0
14:8	ECC_WDataecc5BUS	Eccdata from the register will be written to the RAM.	WO	0x0
6:0	ECC_WDataecc4BUS	Eccdata from the register will be written to the RAM.	WO	0x0

ECC_dbytectrl

ECC Data Byte Control. The maximum number of implemented byte enables equals the data width of the RAM divided by 8.

Module Instance	Base Address	Register Address
ecc_emac0_rx_ecc_registerBlock	0xFF8C0800	0xFF8C0874
ecc_emac1_rx_ecc_registerBlock	0xFF8C1000	0xFF8C1074
ecc_emac2_rx_ecc_registerBlock	0xFF8C1800	0xFF8C1874

Offset: 0x74

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															DBEN 0x0

ECC_dbytectrl Fields

Bit	Name	Description	Access	Reset
0	DBEN	Byte or word enable for access.	RW	0x0

ECC_accctrl

These bits determine which byte of data/ecc to write to RAM.

Module Instance	Base Address	Register Address
ecc_emac0_rx_ecc_registerBlock	0xFF8C0800	0xFF8C0878
ecc_emac1_rx_ecc_registerBlock	0xFF8C1000	0xFF8C1078
ecc_emac2_rx_ecc_registerBlock	0xFF8C1800	0xFF8C1878

Offset: 0x78

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							RDWR 0x0	Reserved						ECCOVR 0x0	DATAOVR 0x0

ECC_accctrl Fields

Bit	Name	Description	Access	Reset
8	RDWR	Control for read/write.	RW	0x0
1	ECCOVR	ECC Data Override.	RW	0x0

Bit	Name	Description	Access	Reset
0	DATAOVR	RAM Data Override. Override the ECC_dataBUS register with RAM data in read mode set by ECC_RW. 1'b0: Data override disabled. 1'b1: Data override enabled.	RW	0x0

ECC_startacc

These bits determine which byte of data/ecc to write to RAM.

Module Instance	Base Address	Register Address
ecc_emac0_rx_ecc_registerBlock	0xFF8C0800	0xFF8C087C
ecc_emac1_rx_ecc_registerBlock	0xFF8C1000	0xFF8C107C
ecc_emac2_rx_ecc_registerBlock	0xFF8C1800	0xFF8C187C

Offset: 0x7C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															ENBUSA 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															

ECC_startacc Fields

Bit	Name	Description	Access	Reset
16	ENBUSA	Start RAM access for PORTA.	RW	0x0

ECC_wdctrl

Bits to Enable/Disable Watch Dog Timer

Module Instance	Base Address	Register Address
ecc_emac0_rx_ecc_registerBlock	0xFF8C0800	0xFF8C0880
ecc_emac1_rx_ecc_registerBlock	0xFF8C1000	0xFF8C1080
ecc_emac2_rx_ecc_registerBlock	0xFF8C1800	0xFF8C1880

Offset: 0x80

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															WDEN_RAM 0x0

ECC_wdctrl Fields

Bit	Name	Description	Access	Reset
0	WDEN_RAM	Enable watchdog timeout for OCP register access to IP RAM.	RW	0x0

SERRLKUPA0

Single-bit error address in LOOKUP TABLE for PORTA.

Module Instance	Base Address	Register Address
ecc_emac0_rx_ecc_registerBlock	0xFF8C0800	0xFF8C0890
ecc_emac1_rx_ecc_registerBlock	0xFF8C1000	0xFF8C1090
ecc_emac2_rx_ecc_registerBlock	0xFF8C1800	0xFF8C1890

Offset: 0x90

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VALID RW 0x0	Reserved														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				Address RO 0x0											

SERRLKUPA0 Fields

Bit	Name	Description	Access	Reset
31	VALID	Valid flag bit. Valid bit indicates if the address in this register is current or stale.	RW	0x0
11:0	Address	Recent Single-bit error address. This register shows the address of the each single-bit error. RAM size will determine the maximum number of address bits. If ram size is 32 Kbytes, bit 30-16 will be reserved and read as zero.	RO	0x0

emac_tx_ecc Address Map

Module Instance	Base Address	End Address
ecc_emac0_tx_ecc_registerBlock	0xFF8C0C00	0xFF8C0FFF
ecc_emac1_tx_ecc_registerBlock	0xFF8C1400	0xFF8C17FF
ecc_emac2_tx_ecc_registerBlock	0xFF8C1C00	0xFF8C1FFF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
IP_REV_ID on page 11-82	0x0	32	RO	0x0	
CTRL on page 11-83	0x8	32	RW	0x0	ECC Control Register

Register	Offset	Width	Access	Reset Value	Description
INITSTAT on page 11-84	0xC	32	RW	0x0	This bit is used to set the initialize the memory and ecc to a known value
ERRINTEN on page 11-85	0x10	32	RW	0x0	Error Interrupt enable
ERRINTENS on page 11-86	0x14	32	RW	0x0	Error Interrupt set
ERRINTENR on page 11-87	0x18	32	RW	0x0	Error Interrupt reset.
INTMODE on page 11-87	0x1C	32	RW	0x0	Reads reflect SERRINTEN.
INTTEST on page 11-88	0x24	32	RW	0x0	This bits is used to test interrupt from ECC RAM to GIC
MODSTAT on page 11-89	0x28	32	RW	0x0	Counter feature status flag
DERRADDRA on page 11-90	0x2C	32	RO	0x0	This register shows the address of PORTA current double-bit error. RAM size will determine the maximum number of address bits.
SERRADDRA on page 11-91	0x30	32	RO	0x0	This register shows the address of PORTA current single-bit error. RAM size will determine the maximum number of address bits.
SERRCNTREG on page 11-91	0x3C	32	RW	0x0	Maximum counter value for single-bit error interrupt
ECC_Addrbus on page 11-92	0x40	32	RW	0x0	MSB bit of address is determined by ADR.
ECC_RData0bus on page 11-93	0x44	32	RO	0x0	Data will be read to this register field.
ECC_RData1bus on page 11-94	0x48	32	RO	0x0	Data will be read to this register field.

Register	Offset	Width	Access	Reset Value	Description
ECC_RData2bus on page 11-94	0x4C	32	RO	0x0	Data will be read to this register field.
ECC_RData3bus on page 11-95	0x50	32	RO	0x0	Data will be read to this register field.
ECC_WData0bus on page 11-96	0x54	32	WO	0x0	Data from the register will be written to the RAM.
ECC_WData1bus on page 11-97	0x58	32	WO	0x0	Data from the register will be written to the RAM.
ECC_WData2bus on page 11-97	0x5C	32	WO	0x0	Data from the register will be written to the RAM.
ECC_WData3bus on page 11-98	0x60	32	WO	0x0	Data from the register will be written to the RAM.
ECC_RDataecc0bus on page 11-99	0x64	32	RO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.
ECC_RDataecc1bus on page 11-100	0x68	32	RO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.
ECC_WDataecc0bus on page 11-101	0x6C	32	WO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Register	Offset	Width	Access	Reset Value	Description
ECC_WDataecc1bus on page 11-102	0x70	32	WO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.
ECC_dbytectrl on page 11-103	0x74	32	RW	0x0	Max number of implemented byte enabled is DAT/8
ECC_accctrl on page 11-104	0x78	32	RW	0x0	These bits determine which byte of data/ecc to write to RAM.
ECC_startacc on page 11-105	0x7C	32	RW	0x0	These bits determine which byte of data/ecc to write to RAM.
ECC_wdctrl on page 11-105	0x80	32	RW	0x0	Bits to Enable/Disable Watch Dog Timer
SERRLKUPA0 on page 11-106	0x90	32	RW	0x0	Single-bit error address in LOOKUP TABLE for PORTA.
INTSTAT on page 11-107	0x	32	RW	0x0	This bit is used to enable interrupt generation on SERR lookup table overflow. When all the entries in the table are valid=1 and this is bit is enabled, serr_req signal will be asserted.

emac_tx_ecc Summary

Module Instance	Base Address
ecc_emac0_tx_ecc_registerBlock	0xFF8C0C00
ecc_emac1_tx_ecc_registerBlock	0xFF8C1400
ecc_emac2_tx_ecc_registerBlock	0xFF8C1C00

Register Address Offset	Bit Fields															
ecc_emac0_tx_ecc_register-Block																
IP_REV_ID 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SIREV RO 0x0															
CTRL 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															INITA 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							CNT_RSTA 0x0	Reserved							ECC_EN 0x0
INITSTAT 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															INITCOM-PLETEA 0x0
ERRINTEN 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTEN 0x0
ERRINTENS 0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTS 0x0
ERRINTENR 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTR 0x0

Register Address Offset	Bit Fields															
INTMODE 0x1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															INTONCMP 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							INTONOVF 0x0	Reserved							INTMODE 0x0
INTTEST 0x24	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							TDERRA 0x0	Reserved							TSERRA 0x0
MODSTAT 0x28	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															CMPFLGA 0x0
DERRADDRA 0x2C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						Address 0x0									
SERRADDRA 0x30	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						Address 0x0									
SERRCNTREG 0x3C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	SERRCNT 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SERRCNT 0x0															

Register Address Offset	Bit Fields															
ECC_Addrbus 0x40	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						ECC_AddrBUS 0x0									
ECC_RData0bus 0x44	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_RData1bus 0x48	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													ECC_RDataBUS 0x0		
ECC_RData2bus 0x4C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_RData3bus 0x50	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_WData0bus 0x54	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															

Register Address Offset	Bit Fields															
ECC_WData1bus 0x58	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													ECC_WDataBUS 0x0		
ECC_WData2bus 0x5C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_WData3bus 0x60	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_RDataecc0bus 0x64	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_RDataecc3BUS 0x0							Reserved	ECC_RDataecc2BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_RDataecc1BUS 0x0							Reserved	ECC_RDataecc0BUS 0x0						
ECC_RDataecc1bus 0x68	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_RDataecc7BUS 0x0							Reserved	ECC_RDataecc6BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_RDataecc5BUS 0x0							Reserved	ECC_RDataecc4BUS 0x0						
ECC_WDataecc0bus 0x6C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_WDataecc3BUS 0x0							Reserved	ECC_WDataecc2BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_WDataecc1BUS 0x0							Reserved	ECC_WDataecc0BUS 0x0						

Register Address Offset	Bit Fields															
ECC_WDataecc1bus 0x70	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_WDataecc7BUS 0x0							Reserved	ECC_WDataecc6BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_WDataecc5BUS 0x0							Reserved	ECC_WDataecc4BUS 0x0						
ECC_dbytectrl 0x74	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															DBEN 0x0
ECC_accctrl 0x78	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							RDWR 0x0	Reserved						ECCOVR 0x0	DATAOVR 0x0
ECC_startacc 0x7C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															ENBUSA 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															
ECC_wdctrl 0x80	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															WDEN_RAM 0x0
SERRLKUPA0 0x90	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	VALID RW 0x0	Reserved														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						Address RO 0x0									

Register Address Offset	Bit Fields															
INTSTAT 0x32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							DERRPENA 0x0	Reserved							SERRPENA 0x0
ecc_emac1_tx_ecc_register-Block																
IP_REV_ID 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SIREV RO 0x0															
CTRL 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															INITA 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							CNT_RSTA 0x0	Reserved							ECC_EN 0x0
INITSTAT 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															INITCOM-PLETEA 0x0
ERRINTEN 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTEN 0x0
ERRINTENS 0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTS 0x0

Register Address Offset	Bit Fields															
ERRINTENR 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTR 0x0
INTMODE 0x1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															INTONCMP 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							INTONOVF 0x0	Reserved							INTMODE 0x0
INTTEST 0x24	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							TDERRA 0x0	Reserved							TSERRA 0x0
MODSTAT 0x28	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															CMPFLGA 0x0
DERRADDRA 0x2C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						Address 0x0									
SERRADDRA 0x30	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						Address 0x0									

Register Address Offset	Bit Fields															
SERRCNTREG 0x3C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	SERRCNT 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SERRCNT 0x0															
ECC_Addrbus 0x40	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						ECC_AddrBUS 0x0									
ECC_RData0bus 0x44	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_RData1bus 0x48	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													ECC_RDataBUS 0x0		
ECC_RData2bus 0x4C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_RData3bus 0x50	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															

Register Address Offset	Bit Fields															
ECC_WData0bus 0x54	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_WData1bus 0x58	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													ECC_WDataBUS 0x0		
ECC_WData2bus 0x5C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_WData3bus 0x60	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_RDataecc0bus 0x64	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_RDataecc3BUS 0x0							Reserved	ECC_RDataecc2BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_RDataecc1BUS 0x0							Reserved	ECC_RDataecc0BUS 0x0						
ECC_RDataecc1bus 0x68	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_RDataecc7BUS 0x0							Reserved	ECC_RDataecc6BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_RDataecc5BUS 0x0							Reserved	ECC_RDataecc4BUS 0x0						

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataecc0bus 0x6C	Reserved	ECC_WDataecc3BUS 0x0							Reserved	ECC_WDataecc2BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_WDataecc1BUS 0x0							Reserved	ECC_WDataecc0BUS 0x0						
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataecc1bus 0x70	Reserved	ECC_WDataecc7BUS 0x0							Reserved	ECC_WDataecc6BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_WDataecc5BUS 0x0							Reserved	ECC_WDataecc4BUS 0x0						
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_dbytectrl 0x74	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															DBEN 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_accctrl 0x78	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							RDWR 0x0	Reserved						ECCOVR 0x0	DATAOVR 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_startacc 0x7C	Reserved															ENBUSA 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_wdctrl 0x80	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															WDEN_RAM 0x0

Register Address Offset	Bit Fields															
SERRLKUPA0 0x90	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	VALID RW 0x0	Reserved														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						Address RO 0x0									
INTSTAT 0x32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							DERRPENA 0x0	Reserved							SERRPENA 0x0
ecc_emac2_tx_ecc_register-Block																
IP_REV_ID 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SIREV RO 0x0															
CTRL 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															INITA 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							CNT_RSTA 0x0	Reserved							ECC_EN 0x0
INITSTAT 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															INITCOMPLETEA 0x0

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
ERRINTEN 0x10	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTEN 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
ERRINTENS 0x14	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTS 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
ERRINTENR 0x18	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTR 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															INTONCMP 0x0
INTMODE 0x1C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							INTONOVF 0x0	Reserved							INTMODE 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
INTTEST 0x24	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							TDERRA 0x0	Reserved							TSERRA 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
MODSTAT 0x28	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															CMPFLGA 0x0

Register Address Offset	Bit Fields															
DERRADDRA 0x2C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						Address 0x0									
SERRADDRA 0x30	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						Address 0x0									
SERRCNTREG 0x3C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	SERRCNT 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SERRCNT 0x0															
ECC_Addrbus 0x40	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						ECC_AddrBUS 0x0									
ECC_RData0bus 0x44	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_RData1bus 0x48	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													ECC_RDataBUS 0x0		

Register Address Offset	Bit Fields															
ECC_RData2bus 0x4C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_RData3bus 0x50	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_WData0bus 0x54	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_WData1bus 0x58	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													ECC_WDataBUS 0x0		
ECC_WData2bus 0x5C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_WData3bus 0x60	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															

Register Address Offset	Bit Fields															
ECC_RDataecc0bus 0x64	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_RDataecc3BUS 0x0							Reserved	ECC_RDataecc2BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_RDataecc1BUS 0x0							Reserved	ECC_RDataecc0BUS 0x0						
ECC_RDataecc1bus 0x68	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_RDataecc7BUS 0x0							Reserved	ECC_RDataecc6BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_RDataecc5BUS 0x0							Reserved	ECC_RDataecc4BUS 0x0						
ECC_WDataecc0bus 0x6C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_WDataecc3BUS 0x0							Reserved	ECC_WDataecc2BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_WDataecc1BUS 0x0							Reserved	ECC_WDataecc0BUS 0x0						
ECC_WDataecc1bus 0x70	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_WDataecc7BUS 0x0							Reserved	ECC_WDataecc6BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_WDataecc5BUS 0x0							Reserved	ECC_WDataecc4BUS 0x0						
ECC_dbytectrl 0x74	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															DBEN 0x0
ECC_accctrl 0x78	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							RDWR 0x0	Reserved						ECCOVR 0x0	DATAOVR 0x0

Register Address Offset	Bit Fields															
ECC_startacc 0x7C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															ENBUSA 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															
ECC_wdctrl 0x80	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															WDEN_RAM 0x0
SERRLKUPA0 0x90	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	VALID RW 0x0	Reserved														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						Address RO 0x0									
INTSTAT 0x32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							DERRPENA 0x0	Reserved							SERRPENA 0x0

IP_REV_ID

Module Instance	Base Address	Register Address
ecc_emac0_tx_ecc_registerBlock	0xFF8C0C00	0xFF8C0C00
ecc_emac1_tx_ecc_registerBlock	0xFF8C1400	0xFF8C1400
ecc_emac2_tx_ecc_registerBlock	0xFF8C1C00	0xFF8C1C00

Offset: 0x0

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SIREV RO 0x0															

IP_REV_ID Fields

Bit	Name	Description	Access	Reset
15:0	SIREV	IP Rev # These bits indicate the silicon revision number.	RO	0x0

CTRL

ECC Control Register

Module Instance	Base Address	Register Address
ecc_emac0_tx_ecc_registerBlock	0xFF8C0C00	0xFF8C0C08
ecc_emac1_tx_ecc_registerBlock	0xFF8C1400	0xFF8C1408
ecc_emac2_tx_ecc_registerBlock	0xFF8C1C00	0xFF8C1C08

Offset: 0x8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															INITA 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							CNT_RSTA 0x0	Reserved							ECC_EN 0x0

CTRL Fields

Bit	Name	Description	Access	Reset
16	INITA	Enable for the hardware memory initialization PORTA.	RW	0x0
8	CNT_RSTA	Enable to reset internal single-bit error counter A value to zero	RW	0x0
0	ECC_EN	Enable for the ECC detection and correction logic.	RW	0x0

INITSTAT

This bit is used to set the initialize the memory and ecc to a known value

Module Instance	Base Address	Register Address
ecc_emac0_tx_ecc_registerBlock	0xFF8C0C00	0xFF8C0C0C
ecc_emac1_tx_ecc_registerBlock	0xFF8C1400	0xFF8C140C
ecc_emac2_tx_ecc_registerBlock	0xFF8C1C00	0xFF8C1C0C

Offset: 0xC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															INITCOM-PLETEA 0x0

INITSTAT Fields

Bit	Name	Description	Access	Reset
0	INITCOMPLETEA	This bit is used to verify if the hardware memory initialization has completed PORTB.	RW	0x0

ERRINTEN

Error Interrupt enable

Module Instance	Base Address	Register Address
ecc_emac0_tx_ecc_registerBlock	0xFF8C0C00	0xFF8C0C10
ecc_emac1_tx_ecc_registerBlock	0xFF8C1400	0xFF8C1410
ecc_emac2_tx_ecc_registerBlock	0xFF8C1C00	0xFF8C1C10

Offset: 0x10

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															SERRINTEN 0x0

ERRINTEN Fields

Bit	Name	Description	Access	Reset
0	SERRINTEN	This bit is used to enable the single bit error interrupt of ECC RAM system	RW	0x0

ERRINTENS

Error Interrupt set

Module Instance	Base Address	Register Address
ecc_emac0_tx_ecc_registerBlock	0xFF8C0C00	0xFF8C0C14
ecc_emac1_tx_ecc_registerBlock	0xFF8C1400	0xFF8C1414
ecc_emac2_tx_ecc_registerBlock	0xFF8C1C00	0xFF8C1C14

Offset: 0x14

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															SERRINTS 0x0

ERRINTENS Fields

Bit	Name	Description	Access	Reset
0	SERRINTS	This bit is used to set the single-bit error interrupt bit.	RW	0x0

ERRINTENR

Error Interrupt reset.

Module Instance	Base Address	Register Address
ecc_emac0_tx_ecc_registerBlock	0xFF8C0C00	0xFF8C0C18
ecc_emac1_tx_ecc_registerBlock	0xFF8C1400	0xFF8C1418
ecc_emac2_tx_ecc_registerBlock	0xFF8C1C00	0xFF8C1C18

Offset: 0x18

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															SERRINTR 0x0

ERRINTENR Fields

Bit	Name	Description	Access	Reset
0	SERRINTR	This bit is used to reset the single-bit error interrupt bit. Reads reflect SERRINTEN. 1'b0: Writing of zero has no effect. 1'b1: By writing one, this bit will reset SERRINTEN bit to 0. This is performing a bitwise writing of this feature.	RW	0x0

INTMODE

Reads reflect SERRINTEN.

Module Instance	Base Address	Register Address
ecc_emac0_tx_ecc_registerBlock	0xFF8C0C00	0xFF8C0C1C
ecc_emac1_tx_ecc_registerBlock	0xFF8C1400	0xFF8C141C
ecc_emac2_tx_ecc_registerBlock	0xFF8C1C00	0xFF8C1C1C

Offset: 0x1C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															INTONCMP 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							INTONOVF 0x0	Reserved							INTMODE 0x0

INTMODE Fields

Bit	Name	Description	Access	Reset
16	INTONCMP	Enable interrupt on compare.	RW	0x0
8	INTONOVF	Enable interrupt on overflow.	RW	0x0
0	INTMODE	Interrupt mode for single-bit errors.	RW	0x0

INTTEST

This bits is used to test interrupt from ECC RAM to GIC

Module Instance	Base Address	Register Address
ecc_emac0_tx_ecc_registerBlock	0xFF8C0C00	0xFF8C0C24
ecc_emac1_tx_ecc_registerBlock	0xFF8C1400	0xFF8C1424
ecc_emac2_tx_ecc_registerBlock	0xFF8C1C00	0xFF8C1C24

Offset: 0x24

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							TDERRA 0x0	Reserved							TSERRA 0x0

INTTEST Fields

Bit	Name	Description	Access	Reset
8	TDERRA	Test PORTA Double-bit error.	RW	0x0
0	TSERRA	Test PORTA Single-bit error.	RW	0x0

MODSTAT

Counter feature status flag

Module Instance	Base Address	Register Address
ecc_emac0_tx_ecc_registerBlock	0xFF8C0C00	0xFF8C0C28
ecc_emac1_tx_ecc_registerBlock	0xFF8C1400	0xFF8C1428
ecc_emac2_tx_ecc_registerBlock	0xFF8C1C00	0xFF8C1C28

Offset: 0x28

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															CMPFLGA 0x0

MODSTAT Fields

Bit	Name	Description	Access	Reset
0	CMPFLGA	Port A compare status flag	RW	0x0

DERRADDRA

This register shows the address of PORTA current double-bit error. RAM size will determine the maximum number of address bits.

Module Instance	Base Address	Register Address
ecc_emac0_tx_ecc_registerBlock	0xFF8C0C00	0xFF8C0C2C
ecc_emac1_tx_ecc_registerBlock	0xFF8C1400	0xFF8C142C
ecc_emac2_tx_ecc_registerBlock	0xFF8C1C00	0xFF8C1C2C

Offset: 0x2C

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						Address 0x0									

DERRADDRA Fields

Bit	Name	Description	Access	Reset
9:0	Address	Recent double-bit error address.	RO	0x0

SERRADDRA

This register shows the address of PORTA current single-bit error. RAM size will determine the maximum number of address bits.

Module Instance	Base Address	Register Address
ecc_emac0_tx_ecc_registerBlock	0xFF8C0C00	0xFF8C0C30
ecc_emac1_tx_ecc_registerBlock	0xFF8C1400	0xFF8C1430
ecc_emac2_tx_ecc_registerBlock	0xFF8C1C00	0xFF8C1C30

Offset: 0x30

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						Address 0x0									

SERRADDRA Fields

Bit	Name	Description	Access	Reset
9:0	Address	Recent single-bit error address.	RO	0x0

SERRCNTREG

Maximum counter value for single-bit error interrupt

Module Instance	Base Address	Register Address
ecc_emac0_tx_ecc_registerBlock	0xFF8C0C00	0xFF8C0C3C
ecc_emac1_tx_ecc_registerBlock	0xFF8C1400	0xFF8C143C
ecc_emac2_tx_ecc_registerBlock	0xFF8C1C00	0xFF8C1C3C

Offset: 0x3C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SERRCNT 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SERRCNT 0x0															

SERRCNTREG Fields

Bit	Name	Description	Access	Reset
31:0	SERRCNT	Counter value	RW	0x0

ECC_Addrbus

MSB bit of address is determined by ADR.

Module Instance	Base Address	Register Address
ecc_emac0_tx_ecc_registerBlock	0xFF8C0C00	0xFF8C0C40
ecc_emac1_tx_ecc_registerBlock	0xFF8C1400	0xFF8C1440
ecc_emac2_tx_ecc_registerBlock	0xFF8C1C00	0xFF8C1C40

Offset: 0x40

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						ECC_AddrBUS 0x0									

ECC_Addrbus Fields

Bit	Name	Description	Access	Reset
9:0	ECC_AddrBUS	Address will be driven to RAM to either read or write the data. Address will be latched by the RAM when the Enbus is asserted.	RW	0x0

ECC_RData0bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_emac0_tx_ecc_registerBlock	0xFF8C0C00	0xFF8C0C44
ecc_emac1_tx_ecc_registerBlock	0xFF8C1400	0xFF8C1444
ecc_emac2_tx_ecc_registerBlock	0xFF8C1C00	0xFF8C1C44

Offset: 0x44

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData0bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_RDataBUS	ECC_RDataBUS[31:0].	RO	0x0

ECC_RData1bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_emac0_tx_ecc_registerBlock	0xFF8C0C00	0xFF8C0C48
ecc_emac1_tx_ecc_registerBlock	0xFF8C1400	0xFF8C1448
ecc_emac2_tx_ecc_registerBlock	0xFF8C1C00	0xFF8C1C48

Offset: 0x48

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													ECC_RDataBUS 0x0		

ECC_RData1bus Fields

Bit	Name	Description	Access	Reset
2:0	ECC_RDataBUS	ECC_RDataBUS[63:32].	RO	0x0

ECC_RData2bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_emac0_tx_ecc_registerBlock	0xFF8C0C00	0xFF8C0C4C
ecc_emac1_tx_ecc_registerBlock	0xFF8C1400	0xFF8C144C
ecc_emac2_tx_ecc_registerBlock	0xFF8C1C00	0xFF8C1C4C

Offset: 0x4C

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData2bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_RDataBUS	ECC_RDataBUS[95:64].	RO	0x0

ECC_RData3bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_emac0_tx_ecc_registerBlock	0xFF8C0C00	0xFF8C0C50
ecc_emac1_tx_ecc_registerBlock	0xFF8C1400	0xFF8C1450
ecc_emac2_tx_ecc_registerBlock	0xFF8C1C00	0xFF8C1C50

Offset: 0x50

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData3bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_RDataBUS	ECC_RDataBUS[127-96].	RO	0x0

ECC_WData0bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_emac0_tx_ecc_registerBlock	0xFF8C0C00	0xFF8C0C54
ecc_emac1_tx_ecc_registerBlock	0xFF8C1400	0xFF8C1454
ecc_emac2_tx_ecc_registerBlock	0xFF8C1C00	0xFF8C1C54

Offset: 0x54

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData0bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_WDataBUS	ECC_WDataBUS[31:0].	WO	0x0

ECC_WData1bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_emac0_tx_ecc_registerBlock	0xFF8C0C00	0xFF8C0C58
ecc_emac1_tx_ecc_registerBlock	0xFF8C1400	0xFF8C1458
ecc_emac2_tx_ecc_registerBlock	0xFF8C1C00	0xFF8C1C58

Offset: 0x58

Access: WO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													ECC_WDataBUS 0x0		

ECC_WData1bus Fields

Bit	Name	Description	Access	Reset
2:0	ECC_WDataBUS	ECC_WDataBUS[63:32].	WO	0x0

ECC_WData2bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_emac0_tx_ecc_registerBlock	0xFF8C0C00	0xFF8C0C5C
ecc_emac1_tx_ecc_registerBlock	0xFF8C1400	0xFF8C145C
ecc_emac2_tx_ecc_registerBlock	0xFF8C1C00	0xFF8C1C5C

Offset: 0x5C

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData2bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_WDataBUS	ECC_WDataBUS[95-64].	WO	0x0

ECC_WData3bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_emac0_tx_ecc_registerBlock	0xFF8C0C00	0xFF8C0C60
ecc_emac1_tx_ecc_registerBlock	0xFF8C1400	0xFF8C1460
ecc_emac2_tx_ecc_registerBlock	0xFF8C1C00	0xFF8C1C60

Offset: 0x60

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData3bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_WDataBUS	ECC_WDataBUS[127-96].	WO	0x0

ECC_RDataecc0bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_emac0_tx_ecc_registerBlock	0xFF8C0C00	0xFF8C0C64
ecc_emac1_tx_ecc_registerBlock	0xFF8C1400	0xFF8C1464
ecc_emac2_tx_ecc_registerBlock	0xFF8C1C00	0xFF8C1C64

Offset: 0x64

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	ECC_RDataecc3BUS 0x0							Reserved	ECC_RDataecc2BUS 0x0						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	ECC_RDataecc1BUS 0x0							Reserved	ECC_RDataecc0BUS 0x0						

ECC_RDataecc0bus Fields

Bit	Name	Description	Access	Reset
30:24	ECC_RDataecc3BUS	Eccdata will be read to this register field.	RO	0x0
22:16	ECC_RDataecc2BUS	Eccdata will be read to this register field.	RO	0x0
14:8	ECC_RDataecc1BUS	Eccdata will be read to this register field.	RO	0x0
6:0	ECC_RDataecc0BUS	Eccdata will be read to this register field.	RO	0x0

ECC_RDataecc1bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_emac0_tx_ecc_registerBlock	0xFF8C0C00	0xFF8C0C68
ecc_emac1_tx_ecc_registerBlock	0xFF8C1400	0xFF8C1468
ecc_emac2_tx_ecc_registerBlock	0xFF8C1C00	0xFF8C1C68

Offset: 0x68

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	ECC_RDataecc7BUS 0x0							Reserved	ECC_RDataecc6BUS 0x0						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	ECC_RDataecc5BUS 0x0							Reserved	ECC_RDataecc4BUS 0x0						

ECC_RDataecc1bus Fields

Bit	Name	Description	Access	Reset
30:24	ECC_RDataecc7BUS	Eccdata will be read to this register field.	RO	0x0
22:16	ECC_RDataecc6BUS	Eccdata will be read to this register field.	RO	0x0
14:8	ECC_RDataecc5BUS	Eccdata will be read to this register field.	RO	0x0
6:0	ECC_RDataecc4BUS	Eccdata will be read to this register field.	RO	0x0

ECC_WDataecc0bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_emac0_tx_ecc_registerBlock	0xFF8C0C00	0xFF8C0C6C
ecc_emac1_tx_ecc_registerBlock	0xFF8C1400	0xFF8C146C
ecc_emac2_tx_ecc_registerBlock	0xFF8C1C00	0xFF8C1C6C

Offset: 0x6C

Access: WO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	ECC_WDataecc3BUS 0x0							Reserved	ECC_WDataecc2BUS 0x0						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	ECC_WDataecc1BUS 0x0							Reserved	ECC_WDataecc0BUS 0x0						

ECC_WDataecc0bus Fields

Bit	Name	Description	Access	Reset
30:24	ECC_WDataecc3BUS	Eccdata from the register will be written to the RAM.	WO	0x0
22:16	ECC_WDataecc2BUS	Eccdata from the register will be written to the RAM.	WO	0x0
14:8	ECC_WDataecc1BUS	Eccdata from the register will be written to the RAM.	WO	0x0
6:0	ECC_WDataecc0BUS	Eccdata from the register will be written to the RAM.	WO	0x0

ECC_WDataecc1bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_emac0_tx_ecc_registerBlock	0xFF8C0C00	0xFF8C0C70
ecc_emac1_tx_ecc_registerBlock	0xFF8C1400	0xFF8C1470
ecc_emac2_tx_ecc_registerBlock	0xFF8C1C00	0xFF8C1C70

Offset: 0x70

Access: WO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	ECC_WDataecc7BUS 0x0							Reserved	ECC_WDataecc6BUS 0x0						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	ECC_WDataecc5BUS 0x0							Reserved	ECC_WDataecc4BUS 0x0						

ECC_WDataecc1bus Fields

Bit	Name	Description	Access	Reset
30:24	ECC_WDataecc7BUS	Eccdata from the register will be written to the RAM.	WO	0x0
22:16	ECC_WDataecc6BUS	Eccdata from the register will be written to the RAM.	WO	0x0
14:8	ECC_WDataecc5BUS	Eccdata from the register will be written to the RAM.	WO	0x0
6:0	ECC_WDataecc4BUS	Eccdata from the register will be written to the RAM.	WO	0x0

ECC_dbytectrl

ECC Data Byte Control. The maximum number of implemented byte enables equals the data width of the RAM divided by 8.

Module Instance	Base Address	Register Address
ecc_emac0_tx_ecc_registerBlock	0xFF8C0C00	0xFF8C0C74
ecc_emac1_tx_ecc_registerBlock	0xFF8C1400	0xFF8C1474
ecc_emac2_tx_ecc_registerBlock	0xFF8C1C00	0xFF8C1C74

Offset: 0x74

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															DBEN 0x0

ECC_dbytectrl Fields

Bit	Name	Description	Access	Reset
0	DBEN	Byte or word enable for access.	RW	0x0

ECC_accctrl

These bits determine which byte of data/ecc to write to RAM.

Module Instance	Base Address	Register Address
ecc_emac0_tx_ecc_registerBlock	0xFF8C0C00	0xFF8C0C78
ecc_emac1_tx_ecc_registerBlock	0xFF8C1400	0xFF8C1478
ecc_emac2_tx_ecc_registerBlock	0xFF8C1C00	0xFF8C1C78

Offset: 0x78

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							RDWR 0x0	Reserved						ECCOVR 0x0	DATAOVR 0x0

ECC_accctrl Fields

Bit	Name	Description	Access	Reset
8	RDWR	Control for read/write.	RW	0x0
1	ECCOVR	ECC Data Override.	RW	0x0

Bit	Name	Description	Access	Reset
0	DATAOVR	RAM Data Override. Override the ECC_dataBUS register with RAM data in read mode set by ECC_RW. 1'b0: Data override disabled. 1'b1: Data override enabled.	RW	0x0

ECC_startacc

These bits determine which byte of data/ecc to write to RAM.

Module Instance	Base Address	Register Address
ecc_emac0_tx_ecc_registerBlock	0xFF8C0C00	0xFF8C0C7C
ecc_emac1_tx_ecc_registerBlock	0xFF8C1400	0xFF8C147C
ecc_emac2_tx_ecc_registerBlock	0xFF8C1C00	0xFF8C1C7C

Offset: 0x7C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															ENBUSA 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															

ECC_startacc Fields

Bit	Name	Description	Access	Reset
16	ENBUSA	Start RAM access for PORTA.	RW	0x0

ECC_wdctrl

Bits to Enable/Disable Watch Dog Timer

Module Instance	Base Address	Register Address
ecc_emac0_tx_ecc_registerBlock	0xFF8C0C00	0xFF8C0C80
ecc_emac1_tx_ecc_registerBlock	0xFF8C1400	0xFF8C1480
ecc_emac2_tx_ecc_registerBlock	0xFF8C1C00	0xFF8C1C80

Offset: 0x80

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															WDEN_RAM 0x0

ECC_wdctrl Fields

Bit	Name	Description	Access	Reset
0	WDEN_RAM	Enable watchdog timeout for OCP register access to IP RAM.	RW	0x0

SERRLKUPA0

Single-bit error address in LOOKUP TABLE for PORTA.

Module Instance	Base Address	Register Address
ecc_emac0_tx_ecc_registerBlock	0xFF8C0C00	0xFF8C0C90
ecc_emac1_tx_ecc_registerBlock	0xFF8C1400	0xFF8C1490
ecc_emac2_tx_ecc_registerBlock	0xFF8C1C00	0xFF8C1C90

Offset: 0x90

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VALID RW 0x0	Reserved														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						Address RO 0x0									

SERRLKUPA0 Fields

Bit	Name	Description	Access	Reset
31	VALID	Valid flag bit. Valid bit indicates if the address in this register is current or stale.	RW	0x0
9:0	Address	Recent Single-bit error address. This register shows the address of the each single-bit error. RAM size will determine the maximum number of address bits. If ram size is 32 Kbytes, bit 30-16 will be reserved and read as zero.	RO	0x0

INTSTAT

This bit is used to enable interrupt generation on SERR lookup table overflow. When all the entries in the table are valid=1 and this is bit is enabled, serr_req signal will be asserted.

Module Instance	Base Address	Register Address
ecc_emac0_tx_ecc_registerBlock	0xFF8C0C00	0xFF8C0C32
ecc_emac1_tx_ecc_registerBlock	0xFF8C1400	0xFF8C1432
ecc_emac2_tx_ecc_registerBlock	0xFF8C1C00	0xFF8C1C32

Offset: 0x32

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							DERRPENA 0x0	Reserved							SERRPENA 0x0

INTSTAT Fields

Bit	Name	Description	Access	Reset
8	DERRPENA	Double-bit error pending for PORTA.	RW	0x0
0	SERRPENA	Single-bit error pending for PORTA.	RW	0x0

nandecc_ecc Address Map

Module Instance	Base Address	End Address
ecc_nandecc_ecc_registerBlock	0xFF8C2000	0xFF8C23FF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
IP_REV_ID on page 11-117	0x0	32	RO	0x0	
CTRL on page 11-117	0x8	32	RW	0x0	ECC Control Register
INITSTAT on page 11-118	0xC	32	RW	0x0	This bit is used to set the initialize the memory and ecc to a known value
ERRINTEN on page 11-119	0x10	32	RW	0x0	Error Interrupt enable
ERRINTENS on page 11-120	0x14	32	RW	0x0	Error Interrupt set

Register	Offset	Width	Access	Reset Value	Description
ERRINTENR on page 11-120	0x18	32	RW	0x0	Error Interrupt reset.
INTMODE on page 11-121	0x1C	32	RW	0x0	Reads reflect SERRINTEN.
INTSTAT on page 11-122	0x20	32	RW	0x0	This bit is used to enable interrupt generation on SERR lookup table overflow. When all the entries in the table are valid=1 and this is bit is enabled, serr_req signal will be asserted.
INTTEST on page 11-123	0x24	32	RW	0x0	This bits is used to test interrupt from ECC RAM to GIC
MODSTAT on page 11-124	0x28	32	RW	0x0	Counter feature status flag
DERRADDRA on page 11-124	0x2C	32	RO	0x0	This register shows the address of PORTA current double-bit error. RAM size will determine the maximum number of address bits.
SERRADDRA on page 11-125	0x30	32	RO	0x0	This register shows the address of PORTA current single-bit error. RAM size will determine the maximum number of address bits.
SERRCNTREG on page 11-126	0x3C	32	RW	0x0	Maximum counter value for single-bit error interrupt
ECC_Addrbus on page 11-127	0x40	32	RW	0x0	MSB bit of address is determined by ADR.
ECC_RData0bus on page 11-127	0x44	32	RO	0x0	Data will be read to this register field.
ECC_RData1bus on page 11-128	0x48	32	RO	0x0	Data will be read to this register field.

Register	Offset	Width	Access	Reset Value	Description
ECC_RData2bus on page 11-129	0x4C	32	RO	0x0	Data will be read to this register field.
ECC_RData3bus on page 11-129	0x50	32	RO	0x0	Data will be read to this register field.
ECC_WData0bus on page 11-130	0x54	32	WO	0x0	Data from the register will be written to the RAM.
ECC_WData1bus on page 11-131	0x58	32	WO	0x0	Data from the register will be written to the RAM.
ECC_WData2bus on page 11-131	0x5C	32	WO	0x0	Data from the register will be written to the RAM.
ECC_WData3bus on page 11-132	0x60	32	WO	0x0	Data from the register will be written to the RAM.
ECC_RDataecc0bus on page 11-133	0x64	32	RO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.
ECC_RDataecc1bus on page 11-133	0x68	32	RO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.
ECC_WDataecc0bus on page 11-134	0x6C	32	WO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Register	Offset	Width	Access	Reset Value	Description
ECC_WDataecc1bus on page 11-135	0x70	32	WO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.
ECC_dbytectrl on page 11-136	0x74	32	RW	0x0	Max number of implemented byte enabled is DAT/8
ECC_accctrl on page 11-137	0x78	32	RW	0x0	These bits determine which byte of data/ecc to write to RAM.
ECC_startacc on page 11-138	0x7C	32	RW	0x0	These bits determine which byte of data/ecc to write to RAM.
ECC_wdctrl on page 11-139	0x80	32	RW	0x0	Bits to Enable/Disable Watch Dog Timer
SERRLKUPA0 on page 11-139	0x90	32	RW	0x0	Single-bit error address in LOOKUP TABLE for PORTA.

nandecc_ecc Summary

Base Address: 0xFF8C2000

Register Address Offset	Bit Fields															
ecc_nandecc_ecc_register-Block																
IP_REV_ID 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SIREV															
	RO 0x0															

Register Address Offset	Bit Fields															
CTRL 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															INITA 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							CNT_RSTA 0x0	Reserved							ECC_EN 0x0
INITSTAT 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															INITCOM-PLETEA 0x0
ERRINTEN 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTEN 0x0
ERRINTENS 0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTS 0x0
ERRINTENR 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTR 0x0
INTMODE 0x1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															INTONCMP 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							INTONOVF 0x0	Reserved							INTMODE 0x0

Register Address Offset	Bit Fields															
INTSTAT 0x20	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							DERRPENA 0x0	Reserved							SERRPENA 0x0
INTTEST 0x24	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							TDERRA 0x0	Reserved							TSERRA 0x0
MODSTAT 0x28	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															CMPFLGA 0x0
DERRADDRA 0x2C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						Address 0x0									
SERRADDRA 0x30	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						Address 0x0									
SERRCNTREG 0x3C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	SERRCNT 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SERRCNT 0x0															

Register Address Offset	Bit Fields															
ECC_Addrbus 0x40	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						ECC_AddrBUS 0x0									
ECC_RData0bus 0x44	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_RData1bus 0x48	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_RData2bus 0x4C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_RData3bus 0x50	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_WData0bus 0x54	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															

Register Address Offset	Bit Fields															
ECC_WData1bus 0x58	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_WData2bus 0x5C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_WData3bus 0x60	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_RDataecc0bus 0x64	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved		ECC_RDataecc3BUS 0x0						Reserved		ECC_RDataecc2BUS 0x0					
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		ECC_RDataecc1BUS 0x0						Reserved		ECC_RDataecc0BUS 0x0					
ECC_RDataecc1bus 0x68	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved		ECC_RDataecc7BUS 0x0						Reserved		ECC_RDataecc6BUS 0x0					
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		ECC_RDataecc5BUS 0x0						Reserved		ECC_RDataecc4BUS 0x0					
ECC_WDataecc0bus 0x6C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved		ECC_WDataecc3BUS 0x0						Reserved		ECC_WDataecc2BUS 0x0					
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		ECC_WDataecc1BUS 0x0						Reserved		ECC_WDataecc0BUS 0x0					

Register Address Offset	Bit Fields															
ECC_WDataecc1bus 0x70	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved		ECC_WDataecc7BUS 0x0						Reserved		ECC_WDataecc6BUS 0x0					
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		ECC_WDataecc5BUS 0x0						Reserved		ECC_WDataecc4BUS 0x0					
ECC_dbytectrl 0x74	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															DBEN 0x0
ECC_accctrl 0x78	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							RDWR 0x0	Reserved						ECCOVR 0x0	DATAOVR 0x0
ECC_startacc 0x7C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															ENBUSA 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															
ECC_wdctrl 0x80	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															WDEN_RAM 0x0
SERRLKUPA0 0x90	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	VALID RW 0x0	Reserved														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						Address RO 0x0									

IP_REV_ID

Module Instance	Base Address	Register Address
ecc_nandecc_ecc_registerBlock	0xFF8C2000	0xFF8C2000

Offset: 0x0

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SIREV RO 0x0															

IP_REV_ID Fields

Bit	Name	Description	Access	Reset
15:0	SIREV	IP Rev # These bits indicate the silicon revision number.	RO	0x0

CTRL

ECC Control Register

Module Instance	Base Address	Register Address
ecc_nandecc_ecc_registerBlock	0xFF8C2000	0xFF8C2008

Offset: 0x8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															INITA 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							CNT_RSTA 0x0	Reserved							ECC_EN 0x0

CTRL Fields

Bit	Name	Description	Access	Reset
16	INITA	Enable for the hardware memory initialization PORTA.	RW	0x0
8	CNT_RSTA	Enable to reset internal single-bit error counter A value to zero	RW	0x0
0	ECC_EN	Enable for the ECC detection and correction logic.	RW	0x0

INITSTAT

This bit is used to set the initialize the memory and ecc to a known value

Module Instance	Base Address	Register Address
ecc_nandecc_ecc_registerBlock	0xFF8C2000	0xFF8C200C

Offset: 0xC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															INITCOM-PLETEA 0x0

INITSTAT Fields

Bit	Name	Description	Access	Reset
0	INITCOMPLETEA	This bit is used to verify if the hardware memory initialization has completed PORTB.	RW	0x0

ERRINTEN

Error Interrupt enable

Module Instance	Base Address	Register Address
ecc_nandecc_ecc_registerBlock	0xFF8C2000	0xFF8C2010

Offset: 0x10

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															SERRINTEN 0x0

ERRINTEN Fields

Bit	Name	Description	Access	Reset
0	SERRINTEN	This bit is used to enable the single bit error interrupt of ECC RAM system	RW	0x0

ERRINTENS

Error Interrupt set

Module Instance	Base Address	Register Address
ecc_nandecc_ecc_registerBlock	0xFF8C2000	0xFF8C2014

Offset: 0x14

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															SERRINTS 0x0

ERRINTENS Fields

Bit	Name	Description	Access	Reset
0	SERRINTS	This bit is used to set the single-bit error interrupt bit.	RW	0x0

ERRINTENR

Error Interrupt reset.

Module Instance	Base Address	Register Address
ecc_nandecc_ecc_registerBlock	0xFF8C2000	0xFF8C2018

Offset: 0x18

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															SERRINTR 0x0

ERRINTENR Fields

Bit	Name	Description	Access	Reset
0	SERRINTR	This bit is used to reset the single-bit error interrupt bit. Reads reflect SERRINTEN. 1'b0: Writing of zero has no effect. 1'b1: By writing one, this bit will reset SERRINTEN bit to 0. This is performing a bitwise writing of this feature.	RW	0x0

INTMODE

Reads reflect SERRINTEN.

Module Instance	Base Address	Register Address
ecc_nandecc_ecc_registerBlock	0xFF8C2000	0xFF8C201C

Offset: 0x1C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															INTONCMP 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							INTONOVF 0x0	Reserved							INTMODE 0x0

INTMODE Fields

Bit	Name	Description	Access	Reset
16	INTONCMP	Enable interrupt on compare.	RW	0x0
8	INTONOVF	Enable interrupt on overflow.	RW	0x0
0	INTMODE	Interrupt mode for single-bit errors.	RW	0x0

INTSTAT

This bit is used to enable interrupt generation on SERR lookup table overflow. When all the entries in the table are valid=1 and this is bit is enabled, serr_req signal will be asserted.

Module Instance	Base Address	Register Address
ecc_nandecc_ecc_registerBlock	0xFF8C2000	0xFF8C2020

Offset: 0x20

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							DERRPENA 0x0	Reserved							SERRPENA 0x0

INTSTAT Fields

Bit	Name	Description	Access	Reset
8	DERRPENA	Double-bit error pending for PORTA.	RW	0x0
0	SERRPENA	Single-bit error pending for PORTA.	RW	0x0

INTTEST

This bits is used to test interrupt from ECC RAM to GIC

Module Instance	Base Address	Register Address
ecc_nandecc_ecc_registerBlock	0xFF8C2000	0xFF8C2024

Offset: 0x24

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							TDERRA 0x0	Reserved							TSERRA 0x0

INTTEST Fields

Bit	Name	Description	Access	Reset
8	TDERRA	Test PORTA Double-bit error.	RW	0x0
0	TSERRA	Test PORTA Single-bit error.	RW	0x0

MODSTAT

Counter feature status flag

Module Instance	Base Address	Register Address
ecc_nandecc_ecc_registerBlock	0xFF8C2000	0xFF8C2028

Offset: 0x28

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															CMPFLGA 0x0

MODSTAT Fields

Bit	Name	Description	Access	Reset
0	CMPFLGA	Port A compare status flag	RW	0x0

DERRADDRA

This register shows the address of PORTA current double-bit error. RAM size will determine the maximum number of address bits.

Module Instance	Base Address	Register Address
ecc_nandecc_ecc_registerBlock	0xFF8C2000	0xFF8C202C

Offset: 0x2C

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						Address 0x0									

DERRADDRA Fields

Bit	Name	Description	Access	Reset
9:0	Address	Recent double-bit error address.	RO	0x0

SERRADDRA

This register shows the address of PORTA current single-bit error. RAM size will determine the maximum number of address bits.

Module Instance	Base Address	Register Address
ecc_nandecc_ecc_registerBlock	0xFF8C2000	0xFF8C2030

Offset: 0x30

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						Address 0x0									

SERRADDRA Fields

Bit	Name	Description	Access	Reset
9:0	Address	Recent single-bit error address.	RO	0x0

SERRCNTREG

Maximum counter value for single-bit error interrupt

Module Instance	Base Address	Register Address
ecc_nandecc_ecc_registerBlock	0xFF8C2000	0xFF8C203C

Offset: 0x3C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SERRCNT 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SERRCNT 0x0															

SERRCNTREG Fields

Bit	Name	Description	Access	Reset
31:0	SERRCNT	Counter value	RW	0x0

ECC_Addrbus

MSB bit of address is determined by ADR.

Module Instance	Base Address	Register Address
ecc_nandecc_ecc_registerBlock	0xFF8C2000	0xFF8C2040

Offset: 0x40

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						ECC_AddrBUS 0x0									

ECC_Addrbus Fields

Bit	Name	Description	Access	Reset
9:0	ECC_AddrBUS	Address will be driven to RAM to either read or write the data. Address will be latched by the RAM when the Enbus is asserted.	RW	0x0

ECC_RData0bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_nandecc_ecc_registerBlock	0xFF8C2000	0xFF8C2044

Offset: 0x44

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData0bus Fields

Bit	Name	Description	Access	Reset
15:0	ECC_RDataBUS	ECC_RDataBUS[31:0].	RO	0x0

ECC_RData1bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_nandecc_ecc_registerBlock	0xFF8C2000	0xFF8C2048

Offset: 0x48

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData1bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_RDataBUS	ECC_RDataBUS[63:32].	RO	0x0

ECC_RData2bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_nandecc_ecc_registerBlock	0xFF8C2000	0xFF8C204C

Offset: 0x4C

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData2bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_RDataBUS	ECC_RDataBUS[95:64].	RO	0x0

ECC_RData3bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_nandecc_ecc_registerBlock	0xFF8C2000	0xFF8C2050

Offset: 0x50

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData3bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_RDataBUS	ECC_RDataBUS[127-96].	RO	0x0

ECC_WData0bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_nandecc_ecc_registerBlock	0xFF8C2000	0xFF8C2054

Offset: 0x54

Access: WO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData0bus Fields

Bit	Name	Description	Access	Reset
15:0	ECC_WDataBUS	ECC_WDataBUS[31:0].	WO	0x0

ECC_WData1bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_nandecc_ecc_registerBlock	0xFF8C2000	0xFF8C2058

Offset: 0x58

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData1bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_WDataBUS	ECC_WDataBUS[63:32].	WO	0x0

ECC_WData2bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_nandecc_ecc_registerBlock	0xFF8C2000	0xFF8C205C

Offset: 0x5C

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData2bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_WDataBUS	ECC_WDataBUS[95-64].	WO	0x0

ECC_WData3bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_nandecc_ecc_registerBlock	0xFF8C2000	0xFF8C2060

Offset: 0x60

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData3bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_WDataBUS	ECC_WDataBUS[127-96].	WO	0x0

ECC_RDataecc0bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_nandecc_ecc_registerBlock	0xFF8C2000	0xFF8C2064

Offset: 0x64

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		ECC_RDataecc3BUS 0x0						Reserved		ECC_RDataecc2BUS 0x0					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		ECC_RDataecc1BUS 0x0						Reserved		ECC_RDataecc0BUS 0x0					

ECC_RDataecc0bus Fields

Bit	Name	Description	Access	Reset
29:24	ECC_RDataecc3BUS	Eccdata will be read to this register field.	RO	0x0
21:16	ECC_RDataecc2BUS	Eccdata will be read to this register field.	RO	0x0
13:8	ECC_RDataecc1BUS	Eccdata will be read to this register field.	RO	0x0
5:0	ECC_RDataecc0BUS	Eccdata will be read to this register field.	RO	0x0

ECC_RDataecc1bus

The msb bit for the register is configured based on DAT

parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_nandecc_ecc_registerBlock	0xFF8C2000	0xFF8C2068

Offset: 0x68

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		ECC_RDataecc7BUS 0x0						Reserved		ECC_RDataecc6BUS 0x0					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		ECC_RDataecc5BUS 0x0						Reserved		ECC_RDataecc4BUS 0x0					

ECC_RDataecc1bus Fields

Bit	Name	Description	Access	Reset
29:24	ECC_RDataecc7BUS	Eccdata will be read to this register field.	RO	0x0
21:16	ECC_RDataecc6BUS	Eccdata will be read to this register field.	RO	0x0
13:8	ECC_RDataecc5BUS	Eccdata will be read to this register field.	RO	0x0
5:0	ECC_RDataecc4BUS	Eccdata will be read to this register field.	RO	0x0

ECC_WDataecc0bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_nandecc_ecc_registerBlock	0xFF8C2000	0xFF8C206C

Offset: 0x6C

Access: WO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		ECC_WDataecc3BUS 0x0						Reserved		ECC_WDataecc2BUS 0x0					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		ECC_WDataecc1BUS 0x0						Reserved		ECC_WDataecc0BUS 0x0					

ECC_WDataecc0bus Fields

Bit	Name	Description	Access	Reset
29:24	ECC_WDataecc3BUS	Eccdata from the register will be written to the RAM.	WO	0x0
21:16	ECC_WDataecc2BUS	Eccdata from the register will be written to the RAM.	WO	0x0
13:8	ECC_WDataecc1BUS	Eccdata from the register will be written to the RAM.	WO	0x0
5:0	ECC_WDataecc0BUS	Eccdata from the register will be written to the RAM.	WO	0x0

ECC_WDataecc1bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_nandecc_ecc_registerBlock	0xFF8C2000	0xFF8C2070

Offset: 0x70

Access: WO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		ECC_WDataecc7BUS 0x0						Reserved		ECC_WDataecc6BUS 0x0					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		ECC_WDataecc5BUS 0x0						Reserved		ECC_WDataecc4BUS 0x0					

ECC_WDataecc1bus Fields

Bit	Name	Description	Access	Reset
29:24	ECC_WDataecc7BUS	Eccdata from the register will be written to the RAM.	WO	0x0
21:16	ECC_WDataecc6BUS	Eccdata from the register will be written to the RAM.	WO	0x0
13:8	ECC_WDataecc5BUS	Eccdata from the register will be written to the RAM.	WO	0x0
5:0	ECC_WDataecc4BUS	Eccdata from the register will be written to the RAM.	WO	0x0

ECC_dbytectrl

ECC Data Byte Control. The maximum number of implemented byte enables equals the data width of the RAM divided by 8.

Module Instance	Base Address	Register Address
ecc_nandecc_ecc_registerBlock	0xFF8C2000	0xFF8C2074

Offset: 0x74

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															DBEN 0x0

ECC_dbytectrl Fields

Bit	Name	Description	Access	Reset
0	DBEN	Byte or word enable for access.	RW	0x0

ECC_accctrl

These bits determine which byte of data/ecc to write to RAM.

Module Instance	Base Address	Register Address
ecc_nandecc_ecc_registerBlock	0xFF8C2000	0xFF8C2078

Offset: 0x78

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							RDWR 0x0	Reserved						ECCOVR 0x0	DATAOVR 0x0

ECC_accctrl Fields

Bit	Name	Description	Access	Reset
8	RDWR	Control for read/write.	RW	0x0
1	ECCOVR	ECC Data Override.	RW	0x0
0	DATAOVR	RAM Data Override. Override the ECC_dataBUS register with RAM data in read mode set by ECC_RW. 1'b0: Data override disabled. 1'b1: Data override enabled.	RW	0x0

ECC_startacc

These bits determine which byte of data/ecc to write to RAM.

Module Instance	Base Address	Register Address
ecc_nandecc_ecc_registerBlock	0xFF8C2000	0xFF8C207C

Offset: 0x7C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															ENBUSA 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															

ECC_startacc Fields

Bit	Name	Description	Access	Reset
16	ENBUSA	Start RAM access for PORTA.	RW	0x0

ECC_wdctrl

Bits to Enable/Disable Watch Dog Timer

Module Instance	Base Address	Register Address
ecc_nandecc_ecc_registerBlock	0xFF8C2000	0xFF8C2080

Offset: 0x80

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															WDEN_RAM 0x0

ECC_wdctrl Fields

Bit	Name	Description	Access	Reset
0	WDEN_RAM	Enable watchdog timeout for OCP register access to IP RAM.	RW	0x0

SERRLKUPA0

Single-bit error address in LOOKUP TABLE for PORTA.

Module Instance	Base Address	Register Address
ecc_nandecc_ecc_registerBlock	0xFF8C2000	0xFF8C2090

Offset: 0x90

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VALID RW 0x0	Reserved														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						Address RO 0x0									

SERRLKUPA0 Fields

Bit	Name	Description	Access	Reset
31	VALID	Valid flag bit. Valid bit indicates if the address in this register is current or stale.	RW	0x0
9:0	Address	Recent Single-bit error address. This register shows the address of the each single-bit error. RAM size will determine the maximum number of address bits. If ram size is 32 Kbytes, bit 30-16 will be reserved and read as zero.	RO	0x0

nandr_ecc Address Map

Module Instance	Base Address	End Address
ecc_nandr_ecc_registerBlock	0xFF8C2400	0xFF8C27FF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
IP_REV_ID on page 11-149	0x0	32	RO	0x0	
CTRL on page 11-150	0x8	32	RW	0x0	ECC Control Register

Register	Offset	Width	Access	Reset Value	Description
INITSTAT on page 11-150	0xC	32	RW	0x0	This bit is used to set the initialize the memory and ecc to a known value
ERRINTEN on page 11-151	0x10	32	RW	0x0	Error Interrupt enable
ERRINTENS on page 11-152	0x14	32	RW	0x0	Error Interrupt set
ERRINTENR on page 11-153	0x18	32	RW	0x0	Error Interrupt reset.
INTMODE on page 11-154	0x1C	32	RW	0x0	Reads reflect SERRINTEN.
INTSTAT on page 11-154	0x20	32	RW	0x0	This bit is used to enable interrupt generation on SERR lookup table overflow. When all the entries in the table are valid=1 and this is bit is enabled, serr_req signal will be asserted.
INTTEST on page 11-155	0x24	32	RW	0x0	This bits is used to test interrupt from ECC RAM to GIC
MODSTAT on page 11-156	0x28	32	RW	0x0	Counter feature status flag
DERRADDRA on page 11-157	0x2C	32	RO	0x0	This register shows the address of PORTA current double-bit error. RAM size will determine the maximum number of address bits.
SERRADDRA on page 11-157	0x30	32	RO	0x0	This register shows the address of PORTA current single-bit error. RAM size will determine the maximum number of address bits.

Register	Offset	Width	Access	Reset Value	Description
SERRCNTREG on page 11-158	0x3C	32	RW	0x0	Maximum counter value for single-bit error interrupt
ECC_Addrbus on page 11-159	0x40	32	RW	0x0	MSB bit of address is determined by ADR.
ECC_RData0bus on page 11-160	0x44	32	RO	0x0	Data will be read to this register field.
ECC_RData1bus on page 11-160	0x48	32	RO	0x0	Data will be read to this register field.
ECC_RData2bus on page 11-161	0x4C	32	RO	0x0	Data will be read to this register field.
ECC_RData3bus on page 11-162	0x50	32	RO	0x0	Data will be read to this register field.
ECC_WData0bus on page 11-162	0x54	32	WO	0x0	Data from the register will be written to the RAM.
ECC_WData1bus on page 11-163	0x58	32	WO	0x0	Data from the register will be written to the RAM.
ECC_WData2bus on page 11-164	0x5C	32	WO	0x0	Data from the register will be written to the RAM.
ECC_WData3bus on page 11-164	0x60	32	WO	0x0	Data from the register will be written to the RAM.
ECC_RDataecc0bus on page 11-165	0x64	32	RO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Register	Offset	Width	Access	Reset Value	Description
ECC_RDataecc1bus on page 11-166	0x68	32	RO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.
ECC_WDataecc0bus on page 11-167	0x6C	32	WO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.
ECC_WDataecc1bus on page 11-168	0x70	32	WO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.
ECC_dbytectrl on page 11-169	0x74	32	RW	0x0	Max number of implemented byte enabled is DAT/8
ECC_accctrl on page 11-170	0x78	32	RW	0x0	These bits determine which byte of data/ecc to write to RAM.
ECC_startacc on page 11-170	0x7C	32	RW	0x0	These bits determine which byte of data/ecc to write to RAM.
ECC_wdctrl on page 11-171	0x80	32	RW	0x0	Bits to Enable/Disable Watch Dog Timer
SERRLKUPA0 on page 11-172	0x90	32	RW	0x0	Single-bit error address in LOOKUP TABLE for PORTA.

nandr_ecc Summary

Base Address: 0xFF8C2400

Register Address Offset	Bit Fields															
ecc_nandr_ecc_register-Block																
IP_REV_ID 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SIREV RO 0x0															
CTRL 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															INITA 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							CNT_RSTA 0x0	Reserved							ECC_EN 0x0
INITSTAT 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															INITCOM-PLETEA 0x0
ERRINTEN 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTEN 0x0
ERRINTENS 0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTS 0x0
ERRINTENR 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTR 0x0

Register Address Offset	Bit Fields															
INTMODE 0x1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															INTONCMP 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							INTONOVF 0x0	Reserved							INTMODE 0x0
INTSTAT 0x20	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							DERRPENA 0x0	Reserved							SERRPENA 0x0
INTTEST 0x24	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							TDERRA 0x0	Reserved							TSERRA 0x0
MODSTAT 0x28	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															CMPFLGA 0x0
DERRADDRA 0x2C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved											Address 0x0				
SERRADDRA 0x30	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved											Address 0x0				

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SERRCNTREG 0x3C	SERRCNT 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SERRCNT 0x0															
ECC_Addrbus 0x40	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved											ECC_AddrBUS 0x0				
ECC_RData0bus 0x44	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_RData1bus 0x48	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_RData2bus 0x4C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_RData3bus 0x50	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															

Register Address Offset	Bit Fields															
ECC_WData0bus 0x54	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_WData1bus 0x58	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_WData2bus 0x5C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_WData3bus 0x60	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_RDataecc0bus 0x64	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_RDataecc3BUS 0x0							Reserved	ECC_RDataecc2BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_RDataecc1BUS 0x0							Reserved	ECC_RDataecc0BUS 0x0						
ECC_RDataecc1bus 0x68	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_RDataecc7BUS 0x0							Reserved	ECC_RDataecc6BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_RDataecc5BUS 0x0							Reserved	ECC_RDataecc4BUS 0x0						

Register Address Offset	Bit Fields															
ECC_WDataecc0bus 0x6C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_WDataecc3BUS 0x0							Reserved	ECC_WDataecc2BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_WDataecc1BUS 0x0							Reserved	ECC_WDataecc0BUS 0x0						
ECC_WDataecc1bus 0x70	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_WDataecc7BUS 0x0							Reserved	ECC_WDataecc6BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_WDataecc5BUS 0x0							Reserved	ECC_WDataecc4BUS 0x0						
ECC_dbytectrl 0x74	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															DBEN 0x0
ECC_accctrl 0x78	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							RDWR 0x0	Reserved						ECCOVR 0x0	DATAOVR 0x0
ECC_startacc 0x7C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															ENBUSA 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															
ECC_wdctrl 0x80	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															WDEN_RAM 0x0

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SERRLKUPA0 0x90	VALID RW 0x0	Reserved														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved											Address RO 0x0				

IP_REV_ID

Module Instance	Base Address	Register Address
ecc_nandr_ecc_registerBlock	0xFF8C2400	0xFF8C2400

Offset: 0x0

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SIREV RO 0x0															

IP_REV_ID Fields

Bit	Name	Description	Access	Reset
15:0	SIREV	IP Rev # These bits indicate the silicon revision number.	RO	0x0

CTRL

ECC Control Register

Module Instance	Base Address	Register Address
ecc_nandr_ecc_registerBlock	0xFF8C2400	0xFF8C2408

Offset: 0x8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															INITA 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							CNT_RSTA 0x0	Reserved							ECC_EN 0x0

CTRL Fields

Bit	Name	Description	Access	Reset
16	INITA	Enable for the hardware memory initialization PORTA.	RW	0x0
8	CNT_RSTA	Enable to reset internal single-bit error counter A value to zero	RW	0x0
0	ECC_EN	Enable for the ECC detection and correction logic.	RW	0x0

INITSTAT

This bit is used to set the initialize the memory and ecc to a known value

Module Instance	Base Address	Register Address
ecc_nandr_ecc_registerBlock	0xFF8C2400	0xFF8C240C

Offset: 0xC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															INITCOM-PLETEA 0x0

INITSTAT Fields

Bit	Name	Description	Access	Reset
0	INITCOMPLETEA	This bit is used to verify if the hardware memory initialization has completed PORTB.	RW	0x0

ERRINTEN

Error Interrupt enable

Module Instance	Base Address	Register Address
ecc_nandr_ecc_registerBlock	0xFF8C2400	0xFF8C2410

Offset: 0x10

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															SERRINTEN 0x0

ERRINTEN Fields

Bit	Name	Description	Access	Reset
0	SERRINTEN	This bit is used to enable the single bit error interrupt of ECC RAM system	RW	0x0

ERRINTENS

Error Interrupt set

Module Instance	Base Address	Register Address
ecc_nandr_ecc_registerBlock	0xFF8C2400	0xFF8C2414

Offset: 0x14

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															SERRINTS 0x0

ERRINTENS Fields

Bit	Name	Description	Access	Reset
0	SERRINTS	This bit is used to set the single-bit error interrupt bit.	RW	0x0

ERRINTENR

Error Interrupt reset.

Module Instance	Base Address	Register Address
ecc_nandr_ecc_registerBlock	0xFF8C2400	0xFF8C2418

Offset: 0x18

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															SERRINTR 0x0

ERRINTENR Fields

Bit	Name	Description	Access	Reset
0	SERRINTR	This bit is used to reset the single-bit error interrupt bit. Reads reflect SERRINTEN. 1'b0: Writing of zero has no effect. 1'b1: By writing one, this bit will reset SERRINTEN bit to 0. This is performing a bitwise writing of this feature.	RW	0x0

INTMODE

Reads reflect SERRINTEN.

Module Instance	Base Address	Register Address
ecc_nandr_ecc_registerBlock	0xFF8C2400	0xFF8C241C

Offset: 0x1C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															INTONCMP 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							INTONOVF 0x0	Reserved							INTMODE 0x0

INTMODE Fields

Bit	Name	Description	Access	Reset
16	INTONCMP	Enable interrupt on compare.	RW	0x0
8	INTONOVF	Enable interrupt on overflow.	RW	0x0
0	INTMODE	Interrupt mode for single-bit errors.	RW	0x0

INTSTAT

This bit is used to enable interrupt generation on SERR lookup table overflow. When all the entries in the table are valid=1 and this is bit is enabled, serr_req signal will be asserted.

Module Instance	Base Address	Register Address
ecc_nandr_ecc_registerBlock	0xFF8C2400	0xFF8C2420

Offset: 0x20

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							DERRPENA 0x0	Reserved							SERRPENA 0x0

INTSTAT Fields

Bit	Name	Description	Access	Reset
8	DERRPENA	Double-bit error pending for PORTA.	RW	0x0
0	SERRPENA	Single-bit error pending for PORTA.	RW	0x0

INTTEST

This bits is used to test interrupt from ECC RAM to GIC

Module Instance	Base Address	Register Address
ecc_nandr_ecc_registerBlock	0xFF8C2400	0xFF8C2424

Offset: 0x24

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							TDERRA 0x0	Reserved							TSERRA 0x0

INTTEST Fields

Bit	Name	Description	Access	Reset
8	TDERRA	Test PORTA Double-bit error.	RW	0x0
0	TSERRA	Test PORTA Single-bit error.	RW	0x0

MODSTAT

Counter feature status flag

Module Instance	Base Address	Register Address
ecc_nandr_ecc_registerBlock	0xFF8C2400	0xFF8C2428

Offset: 0x28

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															CMPFLGA 0x0

MODSTAT Fields

Bit	Name	Description	Access	Reset
0	CMPFLGA	Port A compare status flag	RW	0x0

DERRADDRA

This register shows the address of PORTA current double-bit error. RAM size will determine the maximum number of address bits.

Module Instance	Base Address	Register Address
ecc_nandr_ecc_registerBlock	0xFF8C2400	0xFF8C242C

Offset: 0x2C

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved											Address 0x0				

DERRADDRA Fields

Bit	Name	Description	Access	Reset
4:0	Address	Recent double-bit error address.	RO	0x0

SERRADDRA

This register shows the address of PORTA current single-bit error. RAM size will determine the maximum number of address bits.

Module Instance	Base Address	Register Address
ecc_nandr_ecc_registerBlock	0xFF8C2400	0xFF8C2430

Offset: 0x30

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved											Address 0x0				

SERRADDRA Fields

Bit	Name	Description	Access	Reset
4:0	Address	Recent single-bit error address.	RO	0x0

SERRCNTREG

Maximum counter value for single-bit error interrupt

Module Instance	Base Address	Register Address
ecc_nandr_ecc_registerBlock	0xFF8C2400	0xFF8C243C

Offset: 0x3C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SERRCNT 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SERRCNT 0x0															

SERRCNTREG Fields

Bit	Name	Description	Access	Reset
31:0	SERRCNT	Counter value	RW	0x0

ECC_Addrbus

MSB bit of address is determined by ADR.

Module Instance	Base Address	Register Address
ecc_nandr_ecc_registerBlock	0xFF8C2400	0xFF8C2440

Offset: 0x40

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved											ECC_AddrBUS 0x0				

ECC_Addrbus Fields

Bit	Name	Description	Access	Reset
4:0	ECC_AddrBUS	Address will be driven to RAM to either read or write the data. Address will be latched by the RAM when the Enbus is asserted.	RW	0x0

ECC_RData0bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_nandr_ecc_registerBlock	0xFF8C2400	0xFF8C2444

Offset: 0x44

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData0bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_RDataBUS	ECC_RDataBUS[31:0].	RO	0x0

ECC_RData1bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_nandr_ecc_registerBlock	0xFF8C2400	0xFF8C2448

Offset: 0x48

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData1bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_RDataBUS	ECC_RDataBUS[63:32].	RO	0x0

ECC_RData2bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_nandr_ecc_registerBlock	0xFF8C2400	0xFF8C244C

Offset: 0x4C

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData2bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_RDataBUS	ECC_RDataBUS[95:64].	RO	0x0

ECC_RData3bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_nandr_ecc_registerBlock	0xFF8C2400	0xFF8C2450

Offset: 0x50

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData3bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_RDataBUS	ECC_RDataBUS[127-96].	RO	0x0

ECC_WData0bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_nandr_ecc_registerBlock	0xFF8C2400	0xFF8C2454

Offset: 0x54

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData0bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_WDataBUS	ECC_WDataBUS[31:0].	WO	0x0

ECC_WData1bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_nandr_ecc_registerBlock	0xFF8C2400	0xFF8C2458

Offset: 0x58

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData1bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_WDataBUS	ECC_WDataBUS[63:32].	WO	0x0

ECC_WData2bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_nandr_ecc_registerBlock	0xFF8C2400	0xFF8C245C

Offset: 0x5C

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData2bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_WDataBUS	ECC_WDataBUS[95-64].	WO	0x0

ECC_WData3bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_nandr_ecc_registerBlock	0xFF8C2400	0xFF8C2460

Offset: 0x60

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData3bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_WDataBUS	ECC_WDataBUS[127-96].	WO	0x0

ECC_RDataecc0bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_nandr_ecc_registerBlock	0xFF8C2400	0xFF8C2464

Offset: 0x64

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	ECC_RDataecc3BUS 0x0							Reserved	ECC_RDataecc2BUS 0x0						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	ECC_RDataecc1BUS 0x0							Reserved	ECC_RDataecc0BUS 0x0						

ECC_RDataecc0bus Fields

Bit	Name	Description	Access	Reset
30:24	ECC_RDataecc3BUS	Eccdata will be read to this register field.	RO	0x0
22:16	ECC_RDataecc2BUS	Eccdata will be read to this register field.	RO	0x0
14:8	ECC_RDataecc1BUS	Eccdata will be read to this register field.	RO	0x0
6:0	ECC_RDataecc0BUS	Eccdata will be read to this register field.	RO	0x0

ECC_RDataecc1bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_nandr_ecc_registerBlock	0xFF8C2400	0xFF8C2468

Offset: 0x68

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	ECC_RDataecc7BUS 0x0							Reserved	ECC_RDataecc6BUS 0x0						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	ECC_RDataecc5BUS 0x0							Reserved	ECC_RDataecc4BUS 0x0						

ECC_RDataecc1bus Fields

Bit	Name	Description	Access	Reset
30:24	ECC_RDataecc7BUS	Eccdata will be read to this register field.	RO	0x0
22:16	ECC_RDataecc6BUS	Eccdata will be read to this register field.	RO	0x0
14:8	ECC_RDataecc5BUS	Eccdata will be read to this register field.	RO	0x0
6:0	ECC_RDataecc4BUS	Eccdata will be read to this register field.	RO	0x0

ECC_WDataecc0bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_nandr_ecc_registerBlock	0xFF8C2400	0xFF8C246C

Offset: 0x6C

Access: WO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	ECC_WDataecc3BUS 0x0							Reserved	ECC_WDataecc2BUS 0x0						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	ECC_WDataecc1BUS 0x0							Reserved	ECC_WDataecc0BUS 0x0						

ECC_WDataecc0bus Fields

Bit	Name	Description	Access	Reset
30:24	ECC_WDataecc3BUS	Eccdata from the register will be written to the RAM.	WO	0x0
22:16	ECC_WDataecc2BUS	Eccdata from the register will be written to the RAM.	WO	0x0
14:8	ECC_WDataecc1BUS	Eccdata from the register will be written to the RAM.	WO	0x0
6:0	ECC_WDataecc0BUS	Eccdata from the register will be written to the RAM.	WO	0x0

ECC_WDataecc1bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_nandr_ecc_registerBlock	0xFF8C2400	0xFF8C2470

Offset: 0x70

Access: WO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	ECC_WDataecc7BUS 0x0							Reserved	ECC_WDataecc6BUS 0x0						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	ECC_WDataecc5BUS 0x0							Reserved	ECC_WDataecc4BUS 0x0						

ECC_WDataecc1bus Fields

Bit	Name	Description	Access	Reset
30:24	ECC_WDataecc7BUS	Eccdata from the register will be written to the RAM.	WO	0x0
22:16	ECC_WDataecc6BUS	Eccdata from the register will be written to the RAM.	WO	0x0
14:8	ECC_WDataecc5BUS	Eccdata from the register will be written to the RAM.	WO	0x0
6:0	ECC_WDataecc4BUS	Eccdata from the register will be written to the RAM.	WO	0x0

ECC_dbytectrl

ECC Data Byte Control. The maximum number of implemented byte enables equals the data width of the RAM divided by 8.

Module Instance	Base Address	Register Address
ecc_nandr_ecc_registerBlock	0xFF8C2400	0xFF8C2474

Offset: 0x74

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															DBEN 0x0

ECC_dbytectrl Fields

Bit	Name	Description	Access	Reset
0	DBEN	Byte or word enable for access.	RW	0x0

ECC_accctrl

These bits determine which byte of data/ecc to write to RAM.

Module Instance	Base Address	Register Address
ecc_nandr_ecc_registerBlock	0xFF8C2400	0xFF8C2478

Offset: 0x78

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							RDWR 0x0	Reserved						ECCOVR 0x0	DATAOVR 0x0

ECC_accctrl Fields

Bit	Name	Description	Access	Reset
8	RDWR	Control for read/write.	RW	0x0
1	ECCOVR	ECC Data Override.	RW	0x0
0	DATAOVR	RAM Data Override. Override the ECC_dataBUS register with RAM data in read mode set by ECC_RW. 1'b0: Data override disabled. 1'b1: Data override enabled.	RW	0x0

ECC_startacc

These bits determine which byte of data/ecc to write to RAM.

Module Instance	Base Address	Register Address
ecc_nandr_ecc_registerBlock	0xFF8C2400	0xFF8C247C

Offset: 0x7C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															ENBUSA 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															

ECC_startacc Fields

Bit	Name	Description	Access	Reset
16	ENBUSA	Start RAM access for PORTA.	RW	0x0

ECC_wdctrl

Bits to Enable/Disable Watch Dog Timer

Module Instance	Base Address	Register Address
ecc_nandr_ecc_registerBlock	0xFF8C2400	0xFF8C2480

Offset: 0x80

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															WDEN_RAM 0x0

ECC_wdctrl Fields

Bit	Name	Description	Access	Reset
0	WDEN_RAM	Enable watchdog timeout for OCP register access to IP RAM.	RW	0x0

SERRLKUPA0

Single-bit error address in LOOKUP TABLE for PORTA.

Module Instance	Base Address	Register Address
ecc_nandr_ecc_registerBlock	0xFF8C2400	0xFF8C2490

Offset: 0x90

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VALID RW 0x0	Reserved														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved											Address RO 0x0				

SERRLKUPA0 Fields

Bit	Name	Description	Access	Reset
31	VALID	Valid flag bit. Valid bit indicates if the address in this register is current or stale.	RW	0x0
4:0	Address	Recent Single-bit error address. This register shows the address of the each single-bit error. RAM size will determine the maximum number of address bits. If ram size is 32 Kbytes, bit 30-16 will be reserved and read as zero.	RO	0x0

nandw_ecc Address Map

Module Instance	Base Address	End Address
ecc_nandw_ecc_registerBlock	0xFF8C2800	0xFF8C2BFF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
IP_REV_ID on page 11-181	0x0	32	RO	0x0	
CTRL on page 11-182	0x8	32	RW	0x0	ECC Control Register
INITSTAT on page 11-183	0xC	32	RW	0x0	This bit is used to set the initialize the memory and ecc to a known value
ERRINTEN on page 11-184	0x10	32	RW	0x0	Error Interrupt enable
ERRINTENS on page 11-185	0x14	32	RW	0x0	Error Interrupt set
ERRINTENR on page 11-185	0x18	32	RW	0x0	Error Interrupt reset.

Register	Offset	Width	Access	Reset Value	Description
INTMODE on page 11-186	0x1C	32	RW	0x0	Reads reflect SERRINTEN.
INTSTAT on page 11-187	0x20	32	RW	0x0	This bit is used to enable interrupt generation on SERR lookup table overflow. When all the entries in the table are valid=1 and this is bit is enabled, serr_req signal will be asserted.
INTTEST on page 11-188	0x24	32	RW	0x0	This bits is used to test interrupt from ECC RAM to GIC
MODSTAT on page 11-189	0x28	32	RW	0x0	Counter feature status flag
DERRADDRA on page 11-189	0x2C	32	RO	0x0	This register shows the address of PORTA current double-bit error. RAM size will determine the maximum number of address bits.
SERRADDRA on page 11-190	0x30	32	RO	0x0	This register shows the address of PORTA current single-bit error. RAM size will determine the maximum number of address bits.
SERRCNTREG on page 11-191	0x3C	32	RW	0x0	Maximum counter value for single-bit error interrupt
ECC_Addrbus on page 11-192	0x40	32	RW	0x0	MSB bit of address is determined by ADR.
ECC_RData0bus on page 11-192	0x44	32	RO	0x0	Data will be read to this register field.
ECC_RData1bus on page 11-193	0x48	32	RO	0x0	Data will be read to this register field.
ECC_RData2bus on page 11-194	0x4C	32	RO	0x0	Data will be read to this register field.

Register	Offset	Width	Access	Reset Value	Description
ECC_RData3bus on page 11-194	0x50	32	RO	0x0	Data will be read to this register field.
ECC_WData0bus on page 11-195	0x54	32	WO	0x0	Data from the register will be written to the RAM.
ECC_WData1bus on page 11-196	0x58	32	WO	0x0	Data from the register will be written to the RAM.
ECC_WData2bus on page 11-196	0x5C	32	WO	0x0	Data from the register will be written to the RAM.
ECC_WData3bus on page 11-197	0x60	32	WO	0x0	Data from the register will be written to the RAM.
ECC_RDataecc0bus on page 11-198	0x64	32	RO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.
ECC_RDataecc1bus on page 11-198	0x68	32	RO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.
ECC_WDataecc0bus on page 11-199	0x6C	32	WO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.
ECC_WDataecc1bus on page 11-200	0x70	32	WO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Register	Offset	Width	Access	Reset Value	Description
ECC_dbytectrl on page 11-201	0x74	32	RW	0x0	Max number of implemented byte enabled is DAT/8
ECC_accctrl on page 11-202	0x78	32	RW	0x0	These bits determine which byte of data/ecc to write to RAM.
ECC_startacc on page 11-203	0x7C	32	RW	0x0	These bits determine which byte of data/ecc to write to RAM.
ECC_wdctrl on page 11-204	0x80	32	RW	0x0	Bits to Enable/Disable Watch Dog Timer
SERRLKUPA0 on page 11-204	0x90	32	RW	0x0	Single-bit error address in LOOKUP TABLE for PORTA.

nandw_ecc Summary

Base Address: 0xFF8C2800

Register Address Offset	Bit Fields															
ecc_nandw_ecc_register-Block																
IP_REV_ID 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SIREV RO 0x0															
CTRL 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															INITA 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							CNT_RSTA 0x0	Reserved							ECC_EN 0x0

Register Address Offset	Bit Fields															
INITSTAT 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															INITCOMPLETEA 0x0
ERRINTEN 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTEN 0x0
ERRINTENS 0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTS 0x0
ERRINTENR 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTR 0x0
INTMODE 0x1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															INTONCMP 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							INTONOVF 0x0	Reserved							INTMODE 0x0
INTSTAT 0x20	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							DERRPENA 0x0	Reserved							SERRPENA 0x0

Register Address Offset	Bit Fields															
INTTEST 0x24	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							TDERRA 0x0	Reserved							TSERRA 0x0
MODSTAT 0x28	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															CMPFLGA 0x0
DERRADDRA 0x2C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved									Address 0x0						
SERRADDRA 0x30	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved									Address 0x0						
SERRCNTREG 0x3C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	SERRCNT 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SERRCNT 0x0															
ECC_Addrbus 0x40	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved									ECC_AddrBUS 0x0						

Register Address Offset	Bit Fields															
ECC_RData0bus 0x44	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_RData1bus 0x48	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_RData2bus 0x4C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_RData3bus 0x50	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_WData0bus 0x54	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_WData1bus 0x58	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															

Register Address Offset	Bit Fields															
ECC_WData2bus 0x5C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_WData3bus 0x60	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_RDataec0bus 0x64	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_RDataecc3BUS 0x0							Reserved	ECC_RDataecc2BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_RDataecc1BUS 0x0							Reserved	ECC_RDataecc0BUS 0x0						
ECC_RDataec1bus 0x68	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_RDataecc7BUS 0x0							Reserved	ECC_RDataecc6BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_RDataecc5BUS 0x0							Reserved	ECC_RDataecc4BUS 0x0						
ECC_WDataec0bus 0x6C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_WDataecc3BUS 0x0							Reserved	ECC_WDataecc2BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_WDataecc1BUS 0x0							Reserved	ECC_WDataecc0BUS 0x0						
ECC_WDataec1bus 0x70	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_WDataecc7BUS 0x0							Reserved	ECC_WDataecc6BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_WDataecc5BUS 0x0							Reserved	ECC_WDataecc4BUS 0x0						

Register Address Offset	Bit Fields															
ECC_dbytectrl 0x74	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															DBEN 0x0
ECC_accctrl 0x78	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							RDWR 0x0	Reserved						ECCOVR 0x0	DATAOVR 0x0
ECC_startacc 0x7C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															ENBUSA 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															
ECC_wdctrl 0x80	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															WDEN_RAM 0x0
SERRLKUPA0 0x90	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	VALID RW 0x0	Reserved														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved									Address RO 0x0						

IP_REV_ID

Module Instance	Base Address	Register Address
ecc_nandw_ecc_registerBlock	0xFF8C2800	0xFF8C2800

Offset: 0x0

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SIREV RO 0x0															

IP_REV_ID Fields

Bit	Name	Description	Access	Reset
15:0	SIREV	IP Rev # These bits indicate the silicon revision number.	RO	0x0

CTRL

ECC Control Register

Module Instance	Base Address	Register Address
ecc_nandw_ecc_registerBlock	0xFF8C2800	0xFF8C2808

Offset: 0x8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															INITA 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							CNT_RSTA 0x0	Reserved							ECC_EN 0x0

CTRL Fields

Bit	Name	Description	Access	Reset
16	INITA	Enable for the hardware memory initialization PORTA.	RW	0x0
8	CNT_RSTA	Enable to reset internal single-bit error counter A value to zero	RW	0x0
0	ECC_EN	Enable for the ECC detection and correction logic.	RW	0x0

INITSTAT

This bit is used to set the initialize the memory and ecc to a known value

Module Instance	Base Address	Register Address
ecc_nandw_ecc_registerBlock	0xFF8C2800	0xFF8C280C

Offset: 0xC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															INITCOM-PLETEA 0x0

INITSTAT Fields

Bit	Name	Description	Access	Reset
0	INITCOMPLETEA	This bit is used to verify if the hardware memory initialization has completed PORTB.	RW	0x0

ERRINTEN

Error Interrupt enable

Module Instance	Base Address	Register Address
ecc_nandw_ecc_registerBlock	0xFF8C2800	0xFF8C2810

Offset: 0x10

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															SERRINTEN 0x0

ERRINTEN Fields

Bit	Name	Description	Access	Reset
0	SERRINTEN	This bit is used to enable the single bit error interrupt of ECC RAM system	RW	0x0

ERRINTENS

Error Interrupt set

Module Instance	Base Address	Register Address
ecc_nandw_ecc_registerBlock	0xFF8C2800	0xFF8C2814

Offset: 0x14

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															SERRINTS 0x0

ERRINTENS Fields

Bit	Name	Description	Access	Reset
0	SERRINTS	This bit is used to set the single-bit error interrupt bit.	RW	0x0

ERRINTENR

Error Interrupt reset.

Module Instance	Base Address	Register Address
ecc_nandw_ecc_registerBlock	0xFF8C2800	0xFF8C2818

Offset: 0x18

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															SERRINTR 0x0

ERRINTENR Fields

Bit	Name	Description	Access	Reset
0	SERRINTR	This bit is used to reset the single-bit error interrupt bit. Reads reflect SERRINTEN. 1'b0: Writing of zero has no effect. 1'b1: By writing one, this bit will reset SERRINTEN bit to 0. This is performing a bitwise writing of this feature.	RW	0x0

INTMODE

Reads reflect SERRINTEN.

Module Instance	Base Address	Register Address
ecc_nandw_ecc_registerBlock	0xFF8C2800	0xFF8C281C

Offset: 0x1C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															INTONCMP 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							INTONOVF 0x0	Reserved							INTMODE 0x0

INTMODE Fields

Bit	Name	Description	Access	Reset
16	INTONCMP	Enable interrupt on compare.	RW	0x0
8	INTONOVF	Enable interrupt on overflow.	RW	0x0
0	INTMODE	Interrupt mode for single-bit errors.	RW	0x0

INTSTAT

This bit is used to enable interrupt generation on SERR lookup table overflow. When all the entries in the table are valid=1 and this is bit is enabled, serr_req signal will be asserted.

Module Instance	Base Address	Register Address
ecc_nandw_ecc_registerBlock	0xFF8C2800	0xFF8C2820

Offset: 0x20

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							DERRPENA 0x0	Reserved							SERRPENA 0x0

INTSTAT Fields

Bit	Name	Description	Access	Reset
8	DERRPENA	Double-bit error pending for PORTA.	RW	0x0
0	SERRPENA	Single-bit error pending for PORTA.	RW	0x0

INTTEST

This bits is used to test interrupt from ECC RAM to GIC

Module Instance	Base Address	Register Address
ecc_nandw_ecc_registerBlock	0xFF8C2800	0xFF8C2824

Offset: 0x24

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							TDERRA 0x0	Reserved							TSERRA 0x0

INTTEST Fields

Bit	Name	Description	Access	Reset
8	TDERRA	Test PORTA Double-bit error.	RW	0x0
0	TSERRA	Test PORTA Single-bit error.	RW	0x0

MODSTAT

Counter feature status flag

Module Instance	Base Address	Register Address
ecc_nandw_ecc_registerBlock	0xFF8C2800	0xFF8C2828

Offset: 0x28

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															CMPFLGA 0x0

MODSTAT Fields

Bit	Name	Description	Access	Reset
0	CMPFLGA	Port A compare status flag	RW	0x0

DERRADDRA

This register shows the address of PORTA current double-bit error. RAM size will determine the maximum number of address bits.

Module Instance	Base Address	Register Address
ecc_nandw_ecc_registerBlock	0xFF8C2800	0xFF8C282C

Offset: 0x2C

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									Address 0x0						

DERRADDRA Fields

Bit	Name	Description	Access	Reset
6:0	Address	Recent double-bit error address.	RO	0x0

SERRADDRA

This register shows the address of PORTA current single-bit error. RAM size will determine the maximum number of address bits.

Module Instance	Base Address	Register Address
ecc_nandw_ecc_registerBlock	0xFF8C2800	0xFF8C2830

Offset: 0x30

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									Address 0x0						

SERRADDRA Fields

Bit	Name	Description	Access	Reset
6:0	Address	Recent single-bit error address.	RO	0x0

SERRCNTREG

Maximum counter value for single-bit error interrupt

Module Instance	Base Address	Register Address
ecc_nandw_ecc_registerBlock	0xFF8C2800	0xFF8C283C

Offset: 0x3C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SERRCNT 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SERRCNT 0x0															

SERRCNTREG Fields

Bit	Name	Description	Access	Reset
31:0	SERRCNT	Counter value	RW	0x0

ECC_Addrbus

MSB bit of address is determined by ADR.

Module Instance	Base Address	Register Address
ecc_nandw_ecc_registerBlock	0xFF8C2800	0xFF8C2840

Offset: 0x40

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									ECC_AddrBUS 0x0						

ECC_Addrbus Fields

Bit	Name	Description	Access	Reset
6:0	ECC_AddrBUS	Address will be driven to RAM to either read or write the data. Address will be latched by the RAM when the Enbus is asserted.	RW	0x0

ECC_RData0bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_nandw_ecc_registerBlock	0xFF8C2800	0xFF8C2844

Offset: 0x44

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData0bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_RDataBUS	ECC_RDataBUS[31:0].	RO	0x0

ECC_RData1bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_nandw_ecc_registerBlock	0xFF8C2800	0xFF8C2848

Offset: 0x48

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData1bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_RDataBUS	ECC_RDataBUS[63:32].	RO	0x0

ECC_RData2bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_nandw_ecc_registerBlock	0xFF8C2800	0xFF8C284C

Offset: 0x4C

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData2bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_RDataBUS	ECC_RDataBUS[95:64].	RO	0x0

ECC_RData3bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_nandw_ecc_registerBlock	0xFF8C2800	0xFF8C2850

Offset: 0x50

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData3bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_RDataBUS	ECC_RDataBUS[127-96].	RO	0x0

ECC_WData0bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_nandw_ecc_registerBlock	0xFF8C2800	0xFF8C2854

Offset: 0x54

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData0bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_WDataBUS	ECC_WDataBUS[31:0].	WO	0x0

ECC_WData1bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_nandw_ecc_registerBlock	0xFF8C2800	0xFF8C2858

Offset: 0x58

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData1bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_WDataBUS	ECC_WDataBUS[63:32].	WO	0x0

ECC_WData2bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_nandw_ecc_registerBlock	0xFF8C2800	0xFF8C285C

Offset: 0x5C

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData2bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_WDataBUS	ECC_WDataBUS[95-64].	WO	0x0

ECC_WData3bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_nandw_ecc_registerBlock	0xFF8C2800	0xFF8C2860

Offset: 0x60

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData3bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_WDataBUS	ECC_WDataBUS[127-96].	WO	0x0

ECC_RDataecc0bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_nandw_ecc_registerBlock	0xFF8C2800	0xFF8C2864

Offset: 0x64

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	ECC_RDataecc3BUS 0x0							Reserved	ECC_RDataecc2BUS 0x0						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	ECC_RDataecc1BUS 0x0							Reserved	ECC_RDataecc0BUS 0x0						

ECC_RDataecc0bus Fields

Bit	Name	Description	Access	Reset
30:24	ECC_RDataecc3BUS	Eccdata will be read to this register field.	RO	0x0
22:16	ECC_RDataecc2BUS	Eccdata will be read to this register field.	RO	0x0
14:8	ECC_RDataecc1BUS	Eccdata will be read to this register field.	RO	0x0
6:0	ECC_RDataecc0BUS	Eccdata will be read to this register field.	RO	0x0

ECC_RDataecc1bus

The msb bit for the register is configured based on DAT

parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_nandw_ecc_registerBlock	0xFF8C2800	0xFF8C2868

Offset: 0x68

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	ECC_RDataecc7BUS 0x0							Reserved	ECC_RDataecc6BUS 0x0						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	ECC_RDataecc5BUS 0x0							Reserved	ECC_RDataecc4BUS 0x0						

ECC_RDataecc1bus Fields

Bit	Name	Description	Access	Reset
30:24	ECC_RDataecc7BUS	Eccdata will be read to this register field.	RO	0x0
22:16	ECC_RDataecc6BUS	Eccdata will be read to this register field.	RO	0x0
14:8	ECC_RDataecc5BUS	Eccdata will be read to this register field.	RO	0x0
6:0	ECC_RDataecc4BUS	Eccdata will be read to this register field.	RO	0x0

ECC_WDataecc0bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_nandw_ecc_registerBlock	0xFF8C2800	0xFF8C286C

Offset: 0x6C

Access: WO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	ECC_WDataecc3BUS 0x0							Reserved	ECC_WDataecc2BUS 0x0						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	ECC_WDataecc1BUS 0x0							Reserved	ECC_WDataecc0BUS 0x0						

ECC_WDataecc0bus Fields

Bit	Name	Description	Access	Reset
30:24	ECC_WDataecc3BUS	Eccdata from the register will be written to the RAM.	WO	0x0
22:16	ECC_WDataecc2BUS	Eccdata from the register will be written to the RAM.	WO	0x0
14:8	ECC_WDataecc1BUS	Eccdata from the register will be written to the RAM.	WO	0x0
6:0	ECC_WDataecc0BUS	Eccdata from the register will be written to the RAM.	WO	0x0

ECC_WDataecc1bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_nandw_ecc_registerBlock	0xFF8C2800	0xFF8C2870

Offset: 0x70

Access: WO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	ECC_WDataecc7BUS 0x0							Reserved	ECC_WDataecc6BUS 0x0						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	ECC_WDataecc5BUS 0x0							Reserved	ECC_WDataecc4BUS 0x0						

ECC_WDataecc1bus Fields

Bit	Name	Description	Access	Reset
30:24	ECC_WDataecc7BUS	Eccdata from the register will be written to the RAM.	WO	0x0
22:16	ECC_WDataecc6BUS	Eccdata from the register will be written to the RAM.	WO	0x0
14:8	ECC_WDataecc5BUS	Eccdata from the register will be written to the RAM.	WO	0x0
6:0	ECC_WDataecc4BUS	Eccdata from the register will be written to the RAM.	WO	0x0

ECC_dbytectrl

ECC Data Byte Control. The maximum number of implemented byte enables equals the data width of the RAM divided by 8.

Module Instance	Base Address	Register Address
ecc_nandw_ecc_registerBlock	0xFF8C2800	0xFF8C2874

Offset: 0x74

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															DBEN 0x0

ECC_dbytectrl Fields

Bit	Name	Description	Access	Reset
0	DBEN	Byte or word enable for access.	RW	0x0

ECC_accctrl

These bits determine which byte of data/ecc to write to RAM.

Module Instance	Base Address	Register Address
ecc_nandw_ecc_registerBlock	0xFF8C2800	0xFF8C2878

Offset: 0x78

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							RDWR 0x0	Reserved						ECCOVR 0x0	DATAOVR 0x0

ECC_accctrl Fields

Bit	Name	Description	Access	Reset
8	RDWR	Control for read/write.	RW	0x0
1	ECCOVR	ECC Data Override.	RW	0x0
0	DATAOVR	RAM Data Override. Override the ECC_dataBUS register with RAM data in read mode set by ECC_RW. 1'b0: Data override disabled. 1'b1: Data override enabled.	RW	0x0

ECC_startacc

These bits determine which byte of data/ecc to write to RAM.

Module Instance	Base Address	Register Address
ecc_nandw_ecc_registerBlock	0xFF8C2800	0xFF8C287C

Offset: 0x7C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															ENBUSA 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															

ECC_startacc Fields

Bit	Name	Description	Access	Reset
16	ENBUSA	Start RAM access for PORTA.	RW	0x0

ECC_wdctrl

Bits to Enable/Disable Watch Dog Timer

Module Instance	Base Address	Register Address
ecc_nandw_ecc_registerBlock	0xFF8C2800	0xFF8C2880

Offset: 0x80

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															WDEN_RAM 0x0

ECC_wdctrl Fields

Bit	Name	Description	Access	Reset
0	WDEN_RAM	Enable watchdog timeout for OCP register access to IP RAM.	RW	0x0

SERRLKUPA0

Single-bit error address in LOOKUP TABLE for PORTA.

Module Instance	Base Address	Register Address
ecc_nandw_ecc_registerBlock	0xFF8C2800	0xFF8C2890

Offset: 0x90

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VALID RW 0x0	Reserved														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									Address RO 0x0						

SERRLKUPA0 Fields

Bit	Name	Description	Access	Reset
31	VALID	Valid flag bit. Valid bit indicates if the address in this register is current or stale.	RW	0x0
6:0	Address	Recent Single-bit error address. This register shows the address of the each single-bit error. RAM size will determine the maximum number of address bits. If ram size is 32 Kbytes, bit 30-16 will be reserved and read as zero.	RO	0x0

sdmmc_ecc Address Map

Module Instance	Base Address	End Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2FFF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
IP_REV_ID on page 11-215	0x0	32	RO	0x0	
CTRL on page 11-215	0x8	32	RW	0x0	ECC Control Register

Register	Offset	Width	Access	Reset Value	Description
INITSTAT on page 11-216	0xC	32	RW	0x0	This bit is used to set the initialize the memory and ecc to a known value
ERRINTEN on page 11-217	0x10	32	RW	0x0	Error Interrupt enable
ERRINTENS on page 11-218	0x14	32	RW	0x0	Error Interrupt set
ERRINTENR on page 11-219	0x18	32	RW	0x0	Error Interrupt reset.
INTMODE on page 11-220	0x1C	32	RW	0x0	Reads reflect SERRINTEN.
INTSTAT on page 11-220	0x20	32	RW	0x0	This bit is used to enable interrupt generation on SERR lookup table overflow. When all the entries in the table are valid=1 and this is bit is enabled, serr_req signal will be asserted.
INTTEST on page 11-221	0x24	32	RW	0x0	This bits is used to test interrupt from ECC RAM to GIC
MODSTAT on page 11-222	0x28	32	RW	0x0	Counter feature status flag
DERRADDRA on page 11-223	0x2C	32	RO	0x0	This register shows the address of PORTA current double-bit error. RAM size will determine the maximum number of address bits.
SERRADDRA on page 11-224	0x30	32	RO	0x0	This register shows the address of PORTA current single-bit error. RAM size will determine the maximum number of address bits.

Register	Offset	Width	Access	Reset Value	Description
DERRADDRB on page 11-224	0x34	32	RO	0x0	This register shows the address of PORTB current double-bit error. RAM size will determine the maximum number of address bits.
SERRADDRB on page 11-225	0x38	32	RO	0x0	This register shows the address of PORTB current single-bit error. RAM size will determine the maximum number of address bits.
SERRCNTREG on page 11-226	0x3C	32	RW	0x0	Maximum counter value for single-bit error interrupt
ECC_Addrbus on page 11-227	0x40	32	RW	0x0	MSB bit of address is determined by ADR.
ECC_RData0bus on page 11-227	0x44	32	RO	0x0	Data will be read to this register field.
ECC_RData1bus on page 11-228	0x48	32	RO	0x0	Data will be read to this register field.
ECC_RData2bus on page 11-229	0x4C	32	RO	0x0	Data will be read to this register field.
ECC_RData3bus on page 11-229	0x50	32	RO	0x0	Data will be read to this register field.
ECC_WData0bus on page 11-230	0x54	32	WO	0x0	Data from the register will be written to the RAM.
ECC_WData1bus on page 11-231	0x58	32	WO	0x0	Data from the register will be written to the RAM.
ECC_WData2bus on page 11-231	0x5C	32	WO	0x0	Data from the register will be written to the RAM.
ECC_WData3bus on page 11-232	0x60	32	WO	0x0	Data from the register will be written to the RAM.

Register	Offset	Width	Access	Reset Value	Description
ECC_RDataecc0bus on page 11-233	0x64	32	RO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.
ECC_RDataecc1bus on page 11-233	0x68	32	RO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.
ECC_WDataecc0bus on page 11-234	0x6C	32	WO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.
ECC_WDataecc1bus on page 11-235	0x70	32	WO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.
ECC_dbytectrl on page 11-236	0x74	32	RW	0x0	Max number of implemented byte enabled is DAT/8
ECC_accctrl on page 11-237	0x78	32	RW	0x0	These bits determine which byte of data/ecc to write to RAM.
ECC_startacc on page 11-238	0x7C	32	RW	0x0	These bits determine which byte of data/ecc to write to RAM.
ECC_wdctrl on page 11-239	0x80	32	RW	0x0	Bits to Enable/Disable Watch Dog Timer
SERRLKUPA0 on page 11-239	0x90	32	RW	0x0	Single-bit error address in LOOKUP TABLE for PORTA.

Register	Offset	Width	Access	Reset Value	Description
SERRLKUPB0 on page 11-240	0xD0	32	RW	0x0	Single-bit error address in LOOKUP TABLE for PORTB.

sdmmc_ecc Summary

Base Address: 0xFF8C2C00

Register Address Offset	Bit Fields															
ecc_sdmmc_ecc_register-Block																
IP_REV_ID 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SIREV RO 0x0															
CTRL 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved							INITB 0x0	Reserved							INITA 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						CNT_RSTB 0x0	CNT_RSTA 0x0	Reserved							ECC_EN 0x0
INITSTAT 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							INITCOMPLETEB 0x0	Reserved							INITCOM-PLETEA 0x0

Register Address Offset	Bit Fields															
ERRINTEN 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTEN 0x0
ERRINTENS 0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTS 0x0
ERRINTENR 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTR 0x0
INTMODE 0x1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															INTONCMP 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							INTONOVF 0x0	Reserved							INTMODE 0x0
INTSTAT 0x20	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved							DERRPENB 0x0	Reserved							SERRPENB 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							DERRPENA 0x0	Reserved							SERRPENA 0x0
INTTEST 0x24	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved							TDERRB 0x0	Reserved							TSERRB 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							TDERRA 0x0	Reserved							TSERRA 0x0

Register Address Offset	Bit Fields															
MODSTAT 0x28	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														CMPFLGB 0x0	CMPFLGA 0x0
DERRADDRA 0x2C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						Address 0x0									
SERRADDRA 0x30	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						Address 0x0									
DERRADDRB 0x34	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						Address 0x0									
SERRADDRB 0x38	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						Address 0x0									
SERRCNTREG 0x3C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	SERRCNT 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SERRCNT 0x0															
ECC_Addrbus 0x40	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						ECC_AddrBUS 0x0									

Register Address Offset	Bit Fields															
ECC_RData0bus 0x44	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_RData1bus 0x48	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_RData2bus 0x4C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_RData3bus 0x50	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_WData0bus 0x54	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_WData1bus 0x58	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															

Register Address Offset	Bit Fields															
ECC_WData2bus 0x5C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_WData3bus 0x60	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_RDataec0bus 0x64	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_RDataecc3BUS 0x0							Reserved	ECC_RDataecc2BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_RDataecc1BUS 0x0							Reserved	ECC_RDataecc0BUS 0x0						
ECC_RDataec1bus 0x68	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_RDataecc7BUS 0x0							Reserved	ECC_RDataecc6BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_RDataecc5BUS 0x0							Reserved	ECC_RDataecc4BUS 0x0						
ECC_WDataec0bus 0x6C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_WDataecc3BUS 0x0							Reserved	ECC_WDataecc2BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_WDataecc1BUS 0x0							Reserved	ECC_WDataecc0BUS 0x0						
ECC_WDataec1bus 0x70	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_WDataecc7BUS 0x0							Reserved	ECC_WDataecc6BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_WDataecc5BUS 0x0							Reserved	ECC_WDataecc4BUS 0x0						

Register Address Offset	Bit Fields															
ECC_dbytectrl 0x74	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															DBEN 0x0
ECC_accctrl 0x78	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							RDWR 0x0	Reserved						ECCOVR 0x0	DATAOVR 0x0
ECC_startacc 0x7C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															ENBUSA 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															ENBUSB 0x0
ECC_wdctrl 0x80	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															WDEN_RAM 0x0
SERRLKUPA0 0x90	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	VALID RW 0x0	Reserved														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						Address RO 0x0									
SERRLKUPB0 0xD0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	VALID RW 0x0	Reserved														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						Address RO 0x0									

IP_REV_ID

Module Instance	Base Address	Register Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2C00

Offset: 0x0

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SIREV RO 0x0															

IP_REV_ID Fields

Bit	Name	Description	Access	Reset
15:0	SIREV	IP Rev # These bits indicate the silicon revision number.	RO	0x0

CTRL

ECC Control Register

Module Instance	Base Address	Register Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2C08

Offset: 0x8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							INITB 0x0	Reserved							INITA 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						CNT_RSTB 0x0	CNT_RSTA 0x0	Reserved							ECC_EN 0x0

CTRL Fields

Bit	Name	Description	Access	Reset
24	INITB	Enable for the hardware memory initialization PORTB.	RW	0x0
16	INITA	Enable for the hardware memory initialization PORTA.	RW	0x0
9	CNT_RSTB	Enable to reset internal single-bit error counter B value to zero	RW	0x0
8	CNT_RSTA	Enable to reset internal single-bit error counter A value to zero	RW	0x0
0	ECC_EN	Enable for the ECC detection and correction logic.	RW	0x0

INITSTAT

This bit is used to set the initialize the memory and ecc to a known value

Module Instance	Base Address	Register Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2C0C

Offset: 0xC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							INITCOMPLETEB 0x0	Reserved							INITCOM-PLETEA 0x0

INITSTAT Fields

Bit	Name	Description	Access	Reset
8	INITCOMPLETEB	This bit is used to verify if the hardware memory initialization has completed PORTB.	RW	0x0
0	INITCOMPLETEA	This bit is used to verify if the hardware memory initialization has completed PORTB.	RW	0x0

ERRINTEN

Error Interrupt enable

Module Instance	Base Address	Register Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2C10

Offset: 0x10

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															SERRINTEN 0x0

ERRINTEN Fields

Bit	Name	Description	Access	Reset
0	SERRINTEN	This bit is used to enable the single bit error interrupt of ECC RAM system	RW	0x0

ERRINTENS

Error Interrupt set

Module Instance	Base Address	Register Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2C14

Offset: 0x14

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															SERRINTS 0x0

ERRINTENS Fields

Bit	Name	Description	Access	Reset
0	SERRINTS	This bit is used to set the single-bit error interrupt bit.	RW	0x0

ERRINTENR

Error Interrupt reset.

Module Instance	Base Address	Register Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2C18

Offset: 0x18

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															SERRINTR 0x0

ERRINTENR Fields

Bit	Name	Description	Access	Reset
0	SERRINTR	This bit is used to reset the single-bit error interrupt bit. Reads reflect SERRINTEN. 1'b0: Writing of zero has no effect. 1'b1: By writing one, this bit will reset SERRINTEN bit to 0. This is performing a bitwise writing of this feature.	RW	0x0

INTMODE

Reads reflect SERRINTEN.

Module Instance	Base Address	Register Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2C1C

Offset: 0x1C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															INTONCMP 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							INTONOVF 0x0	Reserved							INTMODE 0x0

INTMODE Fields

Bit	Name	Description	Access	Reset
16	INTONCMP	Enable interrupt on compare.	RW	0x0
8	INTONOVF	Enable interrupt on overflow.	RW	0x0
0	INTMODE	Interrupt mode for single-bit errors.	RW	0x0

INTSTAT

This bit is used to enable interrupt generation on SERR lookup table overflow. When all the entries in the table are valid=1 and this is bit is enabled, serr_req signal will be asserted.

Module Instance	Base Address	Register Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2C20

Offset: 0x20

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							DERRPENB 0x0	Reserved							SERRPENB 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							DERRPENA 0x0	Reserved							SERRPENA 0x0

INTSTAT Fields

Bit	Name	Description	Access	Reset
24	DERRPENB	Double-bit error pending PORTB.	RW	0x0
16	SERRPENB	Single-bit error pending for PORTB.	RW	0x0
8	DERRPENA	Double-bit error pending for PORTA.	RW	0x0
0	SERRPENA	Single-bit error pending for PORTA.	RW	0x0

INTTEST

This bits is used to test interrupt from ECC RAM to GIC

Module Instance	Base Address	Register Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2C24

Offset: 0x24

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							TDERRB 0x0	Reserved							TSERRB 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							TDERRA 0x0	Reserved							TSERRA 0x0

INTTEST Fields

Bit	Name	Description	Access	Reset
24	TDERRB	Test PORTB Double-bit error.	RW	0x0
16	TSERRB	Test PORTB Single-bit error.	RW	0x0
8	TDERRA	Test PORTA Double-bit error.	RW	0x0
0	TSERRA	Test PORTA Single-bit error.	RW	0x0

MODSTAT

Counter feature status flag

Module Instance	Base Address	Register Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2C28

Offset: 0x28

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														CMPFLGB 0x0	CMPFLGA 0x0

MODSTAT Fields

Bit	Name	Description	Access	Reset
1	CMPFLGB	Port B compare status flag	RW	0x0
0	CMPFLGA	Port A compare status flag	RW	0x0

DERRADDRA

This register shows the address of PORTA current double-bit error. RAM size will determine the maximum number of address bits.

Module Instance	Base Address	Register Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2C2C

Offset: 0x2C

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						Address 0x0									

DERRADDRA Fields

Bit	Name	Description	Access	Reset
9:0	Address	Recent double-bit error address.	RO	0x0

SERRADDRA

This register shows the address of PORTA current single-bit error. RAM size will determine the maximum number of address bits.

Module Instance	Base Address	Register Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2C30

Offset: 0x30

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						Address 0x0									

SERRADDRA Fields

Bit	Name	Description	Access	Reset
9:0	Address	Recent single-bit error address.	RO	0x0

DERRADDRB

This register shows the address of PORTB current double-bit error. RAM size will determine the maximum number of address bits.

Module Instance	Base Address	Register Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2C34

Offset: 0x34

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						Address 0x0									

DERRADDRB Fields

Bit	Name	Description	Access	Reset
9:0	Address	Recent double-bit error address.	RO	0x0

SERRADDRB

This register shows the address of PORTB current single-bit error. RAM size will determine the maximum number of address bits.

Module Instance	Base Address	Register Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2C38

Offset: 0x38

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						Address 0x0									

SERRADDRB Fields

Bit	Name	Description	Access	Reset
9:0	Address	Recent single-bit error address.	RO	0x0

SERRCNTREG

Maximum counter value for single-bit error interrupt

Module Instance	Base Address	Register Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2C3C

Offset: 0x3C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SERRCNT 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SERRCNT 0x0															

SERRCNTREG Fields

Bit	Name	Description	Access	Reset
31:0	SERRCNT	Counter value	RW	0x0

ECC_Addrbus

MSB bit of address is determined by ADR.

Module Instance	Base Address	Register Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2C40

Offset: 0x40

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						ECC_AddrBUS 0x0									

ECC_Addrbus Fields

Bit	Name	Description	Access	Reset
9:0	ECC_AddrBUS	Address will be driven to RAM to either read or write the data. Address will be latched by the RAM when the Enbus is asserted.	RW	0x0

ECC_RData0bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2C44

Offset: 0x44

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData0bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_RDataBUS	ECC_RDataBUS[31:0].	RO	0x0

ECC_RData1bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2C48

Offset: 0x48

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData1bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_RDataBUS	ECC_RDataBUS[63:32].	RO	0x0

ECC_RData2bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2C4C

Offset: 0x4C

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData2bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_RDataBUS	ECC_RDataBUS[95:64].	RO	0x0

ECC_RData3bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2C50

Offset: 0x50

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData3bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_RDataBUS	ECC_RDataBUS[127-96].	RO	0x0

ECC_WData0bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2C54

Offset: 0x54

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData0bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_WDataBUS	ECC_WDataBUS[31:0].	WO	0x0

ECC_WData1bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2C58

Offset: 0x58

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData1bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_WDataBUS	ECC_WDataBUS[63:32].	WO	0x0

ECC_WData2bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2C5C

Offset: 0x5C

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData2bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_WDataBUS	ECC_WDataBUS[95-64].	WO	0x0

ECC_WData3bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2C60

Offset: 0x60

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData3bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_WDataBUS	ECC_WDataBUS[127-96].	WO	0x0

ECC_RDataecc0bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2C64

Offset: 0x64

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	ECC_RDataecc3BUS 0x0							Reserved	ECC_RDataecc2BUS 0x0						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	ECC_RDataecc1BUS 0x0							Reserved	ECC_RDataecc0BUS 0x0						

ECC_RDataecc0bus Fields

Bit	Name	Description	Access	Reset
30:24	ECC_RDataecc3BUS	Eccdata will be read to this register field.	RO	0x0
22:16	ECC_RDataecc2BUS	Eccdata will be read to this register field.	RO	0x0
14:8	ECC_RDataecc1BUS	Eccdata will be read to this register field.	RO	0x0
6:0	ECC_RDataecc0BUS	Eccdata will be read to this register field.	RO	0x0

ECC_RDataecc1bus

The msb bit for the register is configured based on DAT

parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2C68

Offset: 0x68

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	ECC_RDataecc7BUS 0x0							Reserved	ECC_RDataecc6BUS 0x0						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	ECC_RDataecc5BUS 0x0							Reserved	ECC_RDataecc4BUS 0x0						

ECC_RDataecc1bus Fields

Bit	Name	Description	Access	Reset
30:24	ECC_RDataecc7BUS	Eccdata will be read to this register field.	RO	0x0
22:16	ECC_RDataecc6BUS	Eccdata will be read to this register field.	RO	0x0
14:8	ECC_RDataecc5BUS	Eccdata will be read to this register field.	RO	0x0
6:0	ECC_RDataecc4BUS	Eccdata will be read to this register field.	RO	0x0

ECC_WDataecc0bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2C6C

Offset: 0x6C

Access: WO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	ECC_WDataecc3BUS 0x0							Reserved	ECC_WDataecc2BUS 0x0						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	ECC_WDataecc1BUS 0x0							Reserved	ECC_WDataecc0BUS 0x0						

ECC_WDataecc0bus Fields

Bit	Name	Description	Access	Reset
30:24	ECC_WDataecc3BUS	Eccdata from the register will be written to the RAM.	WO	0x0
22:16	ECC_WDataecc2BUS	Eccdata from the register will be written to the RAM.	WO	0x0
14:8	ECC_WDataecc1BUS	Eccdata from the register will be written to the RAM.	WO	0x0
6:0	ECC_WDataecc0BUS	Eccdata from the register will be written to the RAM.	WO	0x0

ECC_WDataecc1bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2C70

Offset: 0x70

Access: WO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	ECC_WDataecc7BUS 0x0							Reserved	ECC_WDataecc6BUS 0x0						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	ECC_WDataecc5BUS 0x0							Reserved	ECC_WDataecc4BUS 0x0						

ECC_WDataecc1bus Fields

Bit	Name	Description	Access	Reset
30:24	ECC_WDataecc7BUS	Eccdata from the register will be written to the RAM.	WO	0x0
22:16	ECC_WDataecc6BUS	Eccdata from the register will be written to the RAM.	WO	0x0
14:8	ECC_WDataecc5BUS	Eccdata from the register will be written to the RAM.	WO	0x0
6:0	ECC_WDataecc4BUS	Eccdata from the register will be written to the RAM.	WO	0x0

ECC_dbytectrl

ECC Data Byte Control. The maximum number of implemented byte enables equals the data width of the RAM divided by 8.

Module Instance	Base Address	Register Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2C74

Offset: 0x74

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															DBEN 0x0

ECC_dbytectrl Fields

Bit	Name	Description	Access	Reset
0	DBEN	Byte or word enable for access.	RW	0x0

ECC_accctrl

These bits determine which byte of data/ecc to write to RAM.

Module Instance	Base Address	Register Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2C78

Offset: 0x78

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							RDWR 0x0	Reserved						ECCOVR 0x0	DATAOVR 0x0

ECC_accctrl Fields

Bit	Name	Description	Access	Reset
8	RDWR	Control for read/write.	RW	0x0
1	ECCOVR	ECC Data Override.	RW	0x0
0	DATAOVR	RAM Data Override. Override the ECC_dataBUS register with RAM data in read mode set by ECC_RW. 1'b0: Data override disabled. 1'b1: Data override enabled.	RW	0x0

ECC_startacc

These bits determine which byte of data/ecc to write to RAM.

Module Instance	Base Address	Register Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2C7C

Offset: 0x7C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															ENBUSA 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															ENBUSB 0x0

ECC_startacc Fields

Bit	Name	Description	Access	Reset
16	ENBUSA	Start RAM access for PORTA.	RW	0x0
0	ENBUSB	Start RAM access for PORTB.	RW	0x0

ECC_wdctrl

Bits to Enable/Disable Watch Dog Timer

Module Instance	Base Address	Register Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2C80

Offset: 0x80

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															WDEN_RAM 0x0

ECC_wdctrl Fields

Bit	Name	Description	Access	Reset
0	WDEN_RAM	Enable watchdog timeout for OCP register access to IP RAM.	RW	0x0

SERRLKUPA0

Single-bit error address in LOOKUP TABLE for PORTA.

Module Instance	Base Address	Register Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2C90

Offset: 0x90

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VALID RW 0x0	Reserved														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						Address RO 0x0									

SERRLKUPA0 Fields

Bit	Name	Description	Access	Reset
31	VALID	Valid flag bit. Valid bit indicates if the address in this register is current or stale.	RW	0x0
9:0	Address	Recent Single-bit error address. This register shows the address of the each single-bit error. RAM size will determine the maximum number of address bits. If ram size is 32 Kbytes, bit 30-16 will be reserved and read as zero.	RO	0x0

SERRLKUPB0

Single-bit error address in LOOKUP TABLE for PORTB.

Module Instance	Base Address	Register Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2CD0

Offset: 0xD0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VALID RW 0x0	Reserved														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						Address RO 0x0									

SERRLKUPB0 Fields

Bit	Name	Description	Access	Reset
31	VALID	Valid flag bit. Valid bit indicates if the address in this register is current or stale.	RW	0x0
9:0	Address	Recent Single-bit error address. This register shows the address of the each single-bit error. RAM size will determine the maximum number of address bits. If ram size is 32 Kbytes, bit 30-16 will be reserved and read as zero.	RO	0x0

onchip_ram_ecc Address Map

Module Instance	Base Address	End Address
ecc_onchip_ram_ecc_register-Block	0xFF8C3000	0xFF8C7FFF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
IP_REV_ID on page 11-250	0x0	32	RO	0x0	
CTRL on page 11-251	0x8	32	RW	0x2	ECC Control Register

Register	Offset	Width	Access	Reset Value	Description
INITSTAT on page 11-252	0xC	32	RW	0x0	This bit is used to set the initialize the memory and ecc to a known value
ERRINTEN on page 11-252	0x10	32	RW	0x0	Error Interrupt enable
ERRINTENS on page 11-253	0x14	32	RW	0x0	Error Interrupt set
ERRINTENR on page 11-254	0x18	32	RW	0x0	Error Interrupt reset.
INTMODE on page 11-255	0x1C	32	RW	0x0	Reads reflect SERRINTEN.
INTSTAT on page 11-255	0x20	32	RW	0x0	This bit is used to enable interrupt generation on SERR lookup table overflow. When all the entries in the table are valid=1 and this is bit is enabled, serr_req signal will be asserted.
INTTEST on page 11-256	0x24	32	RW	0x0	This bits is used to test interrupt from ECC RAM to GIC
MODSTAT on page 11-257	0x28	32	RW	0x0	Counter feature status flag
DERRADDRA on page 11-258	0x2C	32	RO	0x0	This register shows the address of PORTA current double-bit error. RAM size will determine the maximum number of address bits.
SERRADDRA on page 11-258	0x30	32	RO	0x0	This register shows the address of PORTA current single-bit error. RAM size will determine the maximum number of address bits.

Register	Offset	Width	Access	Reset Value	Description
SERRCNTREG on page 11-259	0x3C	32	RW	0x0	Maximum counter value for single-bit error interrupt
ECC_Addrbus on page 11-260	0x40	32	RW	0x0	MSB bit of address is determined by ADR.
ECC_RData0bus on page 11-261	0x44	32	RO	0x0	Data will be read to this register field.
ECC_RData1bus on page 11-261	0x48	32	RO	0x0	Data will be read to this register field.
ECC_RData2bus on page 11-262	0x4C	32	RO	0x0	Data will be read to this register field.
ECC_RData3bus on page 11-263	0x50	32	RO	0x0	Data will be read to this register field.
ECC_WData0bus on page 11-263	0x54	32	WO	0x0	Data from the register will be written to the RAM.
ECC_WData1bus on page 11-264	0x58	32	WO	0x0	Data from the register will be written to the RAM.
ECC_WData2bus on page 11-265	0x5C	32	WO	0x0	Data from the register will be written to the RAM.
ECC_WData3bus on page 11-265	0x60	32	WO	0x0	Data from the register will be written to the RAM.
ECC_RDataecc0bus on page 11-266	0x64	32	RO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Register	Offset	Width	Access	Reset Value	Description
ECC_RDataecc1bus on page 11-267	0x68	32	RO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.
ECC_WDataecc0bus on page 11-268	0x6C	32	WO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.
ECC_WDataecc1bus on page 11-269	0x70	32	WO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.
ECC_dbytectrl on page 11-270	0x74	32	RW	0x0	Max number of implemented byte enabled is DAT/8
ECC_accctrl on page 11-271	0x78	32	RW	0x0	These bits determine which byte of data/ecc to write to RAM.
ECC_startacc on page 11-272	0x7C	32	RW	0x0	These bits determine which byte of data/ecc to write to RAM.
ECC_wdctrl on page 11-272	0x80	32	RW	0x0	Bits to Enable/Disable Watch Dog Timer
SERRLKUPA0 on page 11-273	0x90	32	RW	0x0	Single-bit error address in LOOKUP TABLE for PORTA.

onchip_ram_ecc Summary

Base Address: 0xFF8C3000

Register Address Offset	Bit Fields															
ecc_onchip_ram_ecc_register-Block																
IP_REV_ID 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SIREV RO 0x0															
CTRL 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															INITA 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							CNT_RSTA 0x0	Reserved						ECC_SLVERR_DIS 0x1	ECC_EN 0x0
INITSTAT 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															INITCOM-PLETEA 0x0
ERRINTEN 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTEN 0x0
ERRINTENS 0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTS 0x0
ERRINTENR 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTR 0x0

Register Address Offset	Bit Fields															
INTMODE 0x1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															INTONCMP 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							INTONOVF 0x0	Reserved							INTMODE 0x0
INTSTAT 0x20	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							DERRPENA 0x0	Reserved							SERRPENA 0x0
INTTEST 0x24	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							TDERRA 0x0	Reserved							TSERRA 0x0
MODSTAT 0x28	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															CMPFLGA 0x0
DERRADDRA 0x2C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	Address 0x0														
SERRADDRA 0x30	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	Address 0x0														

Register Address Offset	Bit Fields															
SERRCNTREG 0x3C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	SERRCNT 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SERRCNT 0x0															
ECC_Addrbus 0x40	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_AddrBUS 0x0														
ECC_RData0bus 0x44	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_RData1bus 0x48	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_RData2bus 0x4C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_RData3bus 0x50	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															

Register Address Offset	Bit Fields															
ECC_WData0bus 0x54	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_WData1bus 0x58	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_WData2bus 0x5C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_WData3bus 0x60	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_RDataecc0bus 0x64	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved			ECC_RDataecc3BUS 0x0					Reserved			ECC_RDataecc2BUS 0x0				
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved			ECC_RDataecc1BUS 0x0					Reserved			ECC_RDataecc0BUS 0x0				
ECC_RDataecc1bus 0x68	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved			ECC_RDataecc7BUS 0x0					Reserved			ECC_RDataecc6BUS 0x0				
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved			ECC_RDataecc5BUS 0x0					Reserved			ECC_RDataecc4BUS 0x0				

Register Address Offset	Bit Fields															
ECC_WDataecc0bus 0x6C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved			ECC_WDataecc3BUS 0x0					Reserved			ECC_WDataecc2BUS 0x0				
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved			ECC_WDataecc1BUS 0x0					Reserved			ECC_WDataecc0BUS 0x0				
ECC_WDataecc1bus 0x70	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved			ECC_WDataecc7BUS 0x0					Reserved			ECC_WDataecc6BUS 0x0				
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved			ECC_WDataecc5BUS 0x0					Reserved			ECC_WDataecc4BUS 0x0				
ECC_dbytectrl 0x74	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								DBEN 0x0							
ECC_accctrl 0x78	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							RDWR 0x0	Reserved						ECCOVR 0x0	DATAOVR 0x0
ECC_startacc 0x7C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															ENBUSA 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															
ECC_wdctrl 0x80	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															WDEN_RAM 0x0

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SERRLKUPA0 0x90	VALID RW 0x0	Reserved														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	Address RO 0x0														

IP_REV_ID

Module Instance	Base Address	Register Address
ecc_onchip_ram_ecc_register-Block	0xFF8C3000	0xFF8C3000

Offset: 0x0

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SIREV RO 0x0															

IP_REV_ID Fields

Bit	Name	Description	Access	Reset
15:0	SIREV	IP Rev # These bits indicate the silicon revision number.	RO	0x0

CTRL

ECC Control Register

Module Instance	Base Address	Register Address
ecc_onchip_ram_ecc_register-Block	0xFF8C3000	0xFF8C3008

Offset: 0x8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															INITA 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							CNT_RSTA 0x0	Reserved						ECC_SLVERR_DIS 0x1	ECC_EN 0x0

CTRL Fields

Bit	Name	Description	Access	Reset
16	INITA	Enable for the hardware memory initialization PORTA.	RW	0x0
8	CNT_RSTA	Enable to reset internal single-bit error counter A value to zero	RW	0x0
1	ECC_SLVERR_DIS	Enable to prevent double-bit ECC error from triggering SLVERR at ONCHIP_RAM AXI Interface RRESP output. Enable = 1'b1 Disable = 1'b0	RW	0x1
0	ECC_EN	Enable for the ECC detection and correction logic.	RW	0x0

INITSTAT

This bit is used to set the initialize the memory and ecc to a known value

Module Instance	Base Address	Register Address
ecc_onchip_ram_ecc_registerBlock	0xFF8C3000	0xFF8C300C

Offset: 0xC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															INITCOMPLETEA 0x0

INITSTAT Fields

Bit	Name	Description	Access	Reset
0	INITCOMPLETEA	This bit is used to verify if the hardware memory initialization has completed PORTB.	RW	0x0

ERRINTEN

Error Interrupt enable

Module Instance	Base Address	Register Address
ecc_onchip_ram_ecc_registerBlock	0xFF8C3000	0xFF8C3010

Offset: 0x10

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															SERRINTEN 0x0

ERRINTEN Fields

Bit	Name	Description	Access	Reset
0	SERRINTEN	This bit is used to enable the single bit error interrupt of ECC RAM system	RW	0x0

ERRINTENS

Error Interrupt set

Module Instance	Base Address	Register Address
ecc_onchip_ram_ecc_register-Block	0xFF8C3000	0xFF8C3014

Offset: 0x14

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															SERRINTS 0x0

ERRINTENS Fields

Bit	Name	Description	Access	Reset
0	SERRINTS	This bit is used to set the single-bit error interrupt bit.	RW	0x0

ERRINTENR

Error Interrupt reset.

Module Instance	Base Address	Register Address
ecc_onchip_ram_ecc_register-Block	0xFF8C3000	0xFF8C3018

Offset: 0x18

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															SERRINTR 0x0

ERRINTENR Fields

Bit	Name	Description	Access	Reset
0	SERRINTR	This bit is used to reset the single-bit error interrupt bit. Reads reflect SERRINTEN. 1'b0: Writing of zero has no effect. 1'b1: By writing one, this bit will reset SERRINTEN bit to 0. This is performing a bitwise writing of this feature.	RW	0x0

INTMODE

Reads reflect SERRINTEN.

Module Instance	Base Address	Register Address
ecc_onchip_ram_ecc_register-Block	0xFF8C3000	0xFF8C301C

Offset: 0x1C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															INTONCMP 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							INTONOVF 0x0	Reserved							INTMODE 0x0

INTMODE Fields

Bit	Name	Description	Access	Reset
16	INTONCMP	Enable interrupt on compare.	RW	0x0
8	INTONOVF	Enable interrupt on overflow.	RW	0x0
0	INTMODE	Interrupt mode for single-bit errors.	RW	0x0

INTSTAT

This bit is used to enable interrupt generation on SERR lookup table overflow. When all the entries in the table are valid=1 and this is bit is enabled, serr_req signal will be asserted.

Module Instance	Base Address	Register Address
ecc_onchip_ram_ecc_register-Block	0xFF8C3000	0xFF8C3020

Offset: 0x20

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							DERRPENA 0x0	Reserved							SERRPENA 0x0

INTSTAT Fields

Bit	Name	Description	Access	Reset
8	DERRPENA	Double-bit error pending for PORTA.	RW	0x0
0	SERRPENA	Single-bit error pending for PORTA.	RW	0x0

INTTEST

This bits is used to test interrupt from ECC RAM to GIC

Module Instance	Base Address	Register Address
ecc_onchip_ram_ecc_register-Block	0xFF8C3000	0xFF8C3024

Offset: 0x24

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							TDERRA 0x0	Reserved							TSERRA 0x0

INTTEST Fields

Bit	Name	Description	Access	Reset
8	TDERRA	Test PORTA Double-bit error.	RW	0x0
0	TSERRA	Test PORTA Single-bit error.	RW	0x0

MODSTAT

Counter feature status flag

Module Instance	Base Address	Register Address
ecc_onchip_ram_ecc_register-Block	0xFF8C3000	0xFF8C3028

Offset: 0x28

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															CMPFLGA 0x0

MODSTAT Fields

Bit	Name	Description	Access	Reset
0	CMPFLGA	Port A compare status flag	RW	0x0

DERRADDRA

This register shows the address of PORTA current double-bit error. RAM size will determine the maximum number of address bits.

Module Instance	Base Address	Register Address
ecc_onchip_ram_ecc_register-Block	0xFF8C3000	0xFF8C302C

Offset: 0x2C

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	Address 0x0														

DERRADDRA Fields

Bit	Name	Description	Access	Reset
14:0	Address	Recent double-bit error address.	RO	0x0

SERRADDRA

This register shows the address of PORTA current single-bit error. RAM size will determine the maximum number of address bits.

Module Instance	Base Address	Register Address
ecc_onchip_ram_ecc_register-Block	0xFF8C3000	0xFF8C3030

Offset: 0x30

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	Address 0x0														

SERRADDRA Fields

Bit	Name	Description	Access	Reset
14:0	Address	Recent single-bit error address.	RO	0x0

SERRCNTREG

Maximum counter value for single-bit error interrupt

Module Instance	Base Address	Register Address
ecc_onchip_ram_ecc_register-Block	0xFF8C3000	0xFF8C303C

Offset: 0x3C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SERRCNT 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SERRCNT 0x0															

SERRCNTREG Fields

Bit	Name	Description	Access	Reset
31:0	SERRCNT	Counter value	RW	0x0

ECC_Addrbus

MSB bit of address is determined by ADR.

Module Instance	Base Address	Register Address
ecc_onchip_ram_ecc_register-Block	0xFF8C3000	0xFF8C3040

Offset: 0x40

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	ECC_AddrBUS 0x0														

ECC_Addrbus Fields

Bit	Name	Description	Access	Reset
14:0	ECC_AddrBUS	Address will be driven to RAM to either read or write the data. Address will be latched by the RAM when the Enbus is asserted.	RW	0x0

ECC_RData0bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_onchip_ram_ecc_register-Block	0xFF8C3000	0xFF8C3044

Offset: 0x44

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData0bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_RDataBUS	ECC_RDataBUS[31:0].	RO	0x0

ECC_RData1bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_onchip_ram_ecc_register-Block	0xFF8C3000	0xFF8C3048

Offset: 0x48

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData1bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_RDataBUS	ECC_RDataBUS[63:32].	RO	0x0

ECC_RData2bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_onchip_ram_ecc_register-Block	0xFF8C3000	0xFF8C304C

Offset: 0x4C

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData2bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_RDataBUS	ECC_RDataBUS[95:64].	RO	0x0

ECC_RData3bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_onchip_ram_ecc_register-Block	0xFF8C3000	0xFF8C3050

Offset: 0x50

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData3bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_RDataBUS	ECC_RDataBUS[127-96].	RO	0x0

ECC_WData0bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_onchip_ram_ecc_register-Block	0xFF8C3000	0xFF8C3054

Offset: 0x54

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData0bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_WDataBUS	ECC_WDataBUS[31:0].	WO	0x0

ECC_WData1bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_onchip_ram_ecc_register-Block	0xFF8C3000	0xFF8C3058

Offset: 0x58

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData1bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_WDataBUS	ECC_WDataBUS[63:32].	WO	0x0

ECC_WData2bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_onchip_ram_ecc_register-Block	0xFF8C3000	0xFF8C305C

Offset: 0x5C

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData2bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_WDataBUS	ECC_WDataBUS[95-64].	WO	0x0

ECC_WData3bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_onchip_ram_ecc_register-Block	0xFF8C3000	0xFF8C3060

Offset: 0x60

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData3bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_WDataBUS	ECC_WDataBUS[127-96].	WO	0x0

ECC_RDataecc0bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_onchip_ram_ecc_register-Block	0xFF8C3000	0xFF8C3064

Offset: 0x64

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved			ECC_RDataecc3BUS 0x0					Reserved			ECC_RDataecc2BUS 0x0				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			ECC_RDataecc1BUS 0x0					Reserved			ECC_RDataecc0BUS 0x0				

ECC_RDataecc0bus Fields

Bit	Name	Description	Access	Reset
28:24	ECC_RDataecc3BUS	Eccdata will be read to this register field.	RO	0x0
20:16	ECC_RDataecc2BUS	Eccdata will be read to this register field.	RO	0x0
12:8	ECC_RDataecc1BUS	Eccdata will be read to this register field.	RO	0x0
4:0	ECC_RDataecc0BUS	Eccdata will be read to this register field.	RO	0x0

ECC_RDataecc1bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_onchip_ram_ecc_register-Block	0xFF8C3000	0xFF8C3068

Offset: 0x68

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved			ECC_RDataecc7BUS 0x0					Reserved			ECC_RDataecc6BUS 0x0				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			ECC_RDataecc5BUS 0x0					Reserved			ECC_RDataecc4BUS 0x0				

ECC_RDataecc1bus Fields

Bit	Name	Description	Access	Reset
28:24	ECC_RDataecc7BUS	Eccdata will be read to this register field.	RO	0x0
20:16	ECC_RDataecc6BUS	Eccdata will be read to this register field.	RO	0x0
12:8	ECC_RDataecc5BUS	Eccdata will be read to this register field.	RO	0x0
4:0	ECC_RDataecc4BUS	Eccdata will be read to this register field.	RO	0x0

ECC_WDataecc0bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_onchip_ram_ecc_register-Block	0xFF8C3000	0xFF8C306C

Offset: 0x6C

Access: WO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved			ECC_WDataecc3BUS 0x0					Reserved			ECC_WDataecc2BUS 0x0				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			ECC_WDataecc1BUS 0x0					Reserved			ECC_WDataecc0BUS 0x0				

ECC_WDataecc0bus Fields

Bit	Name	Description	Access	Reset
28:24	ECC_WDataecc3BUS	Eccdata from the register will be written to the RAM.	WO	0x0
20:16	ECC_WDataecc2BUS	Eccdata from the register will be written to the RAM.	WO	0x0
12:8	ECC_WDataecc1BUS	Eccdata from the register will be written to the RAM.	WO	0x0
4:0	ECC_WDataecc0BUS	Eccdata from the register will be written to the RAM.	WO	0x0

ECC_WDataecc1bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_onchip_ram_ecc_register-Block	0xFF8C3000	0xFF8C3070

Offset: 0x70

Access: WO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved			ECC_WDataecc7BUS 0x0					Reserved			ECC_WDataecc6BUS 0x0				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			ECC_WDataecc5BUS 0x0					Reserved			ECC_WDataecc4BUS 0x0				

ECC_WDataecc1bus Fields

Bit	Name	Description	Access	Reset
28:24	ECC_WDataecc7BUS	Eccdata from the register will be written to the RAM.	WO	0x0
20:16	ECC_WDataecc6BUS	Eccdata from the register will be written to the RAM.	WO	0x0
12:8	ECC_WDataecc5BUS	Eccdata from the register will be written to the RAM.	WO	0x0
4:0	ECC_WDataecc4BUS	Eccdata from the register will be written to the RAM.	WO	0x0

ECC_dbytectrl

ECC Data Byte Control. The maximum number of implemented byte enables equals the data width of the RAM divided by 8.

Module Instance	Base Address	Register Address
ecc_onchip_ram_ecc_register-Block	0xFF8C3000	0xFF8C3074

Offset: 0x74

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								DBEN 0x0							

ECC_dbytectrl Fields

Bit	Name	Description	Access	Reset
7:0	DBEN	Byte or word enable for access.	RW	0x0

ECC_accctrl

These bits determine which byte of data/ecc to write to RAM.

Module Instance	Base Address	Register Address
ecc_onchip_ram_ecc_register-Block	0xFF8C3000	0xFF8C3078

Offset: 0x78

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							RDWR 0x0	Reserved						ECCOVR 0x0	DATAOVR 0x0

ECC_accctrl Fields

Bit	Name	Description	Access	Reset
8	RDWR	Control for read/write.	RW	0x0
1	ECCOVR	ECC Data Override.	RW	0x0
0	DATAOVR	RAM Data Override. Override the ECC_dataBUS register with RAM data in read mode set by ECC_RW. 1'b0: Data override disabled. 1'b1: Data override enabled.	RW	0x0

ECC_startacc

These bits determine which byte of data/ecc to write to RAM.

Module Instance	Base Address	Register Address
ecc_onchip_ram_ecc_register-Block	0xFF8C3000	0xFF8C307C

Offset: 0x7C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															ENBUSA 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															

ECC_startacc Fields

Bit	Name	Description	Access	Reset
16	ENBUSA	Start RAM access for PORTA.	RW	0x0

ECC_wdctrl

Bits to Enable/Disable Watch Dog Timer

Module Instance	Base Address	Register Address
ecc_onchip_ram_ecc_register-Block	0xFF8C3000	0xFF8C3080

Offset: 0x80

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															WDEN_RAM 0x0

ECC_wdctrl Fields

Bit	Name	Description	Access	Reset
0	WDEN_RAM	Enable watchdog timeout for OCP register access to IP RAM.	RW	0x0

SERRLKUPA0

Single-bit error address in LOOKUP TABLE for PORTA.

Module Instance	Base Address	Register Address
ecc_onchip_ram_ecc_register-Block	0xFF8C3000	0xFF8C3090

Offset: 0x90

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VALID RW 0x0	Reserved														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	Address RO 0x0														

SERRLKUPA0 Fields

Bit	Name	Description	Access	Reset
31	VALID	Valid flag bit. Valid bit indicates if the address in this register is current or stale.	RW	0x0
14:0	Address	Recent Single-bit error address. This register shows the address of the each single-bit error. RAM size will determine the maximum number of address bits. If ram size is 32 Kbytes, bit 30-16 will be reserved and read as zero.	RO	0x0

dmac_ecc Address Map

Module Instance	Base Address	End Address
ecc_dmac_ecc_registerBlock	0xFF8C8000	0xFF8C83FF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
IP_REV_ID on page 11-282	0x0	32	RO	0x0	
CTRL on page 11-283	0x8	32	RW	0x0	ECC Control Register
INITSTAT on page 11-284	0xC	32	RW	0x0	This bit is used to set the initialize the memory and ecc to a known value
ERRINTEN on page 11-285	0x10	32	RW	0x0	Error Interrupt enable
ERRINTENS on page 11-286	0x14	32	RW	0x0	Error Interrupt set
ERRINTENR on page 11-286	0x18	32	RW	0x0	Error Interrupt reset.

Register	Offset	Width	Access	Reset Value	Description
INTMODE on page 11-287	0x1C	32	RW	0x0	Reads reflect SERRINTEN.
INTSTAT on page 11-288	0x20	32	RW	0x0	This bit is used to enable interrupt generation on SERR lookup table overflow. When all the entries in the table are valid=1 and this is bit is enabled, serr_req signal will be asserted.
INTTEST on page 11-289	0x24	32	RW	0x0	This bits is used to test interrupt from ECC RAM to GIC
MODSTAT on page 11-290	0x28	32	RW	0x0	Counter feature status flag
DERRADDRA on page 11-290	0x2C	32	RO	0x0	This register shows the address of PORTA current double-bit error. RAM size will determine the maximum number of address bits.
SERRADDRA on page 11-291	0x30	32	RO	0x0	This register shows the address of PORTA current single-bit error. RAM size will determine the maximum number of address bits.
SERRCNTREG on page 11-292	0x3C	32	RW	0x0	Maximum counter value for single-bit error interrupt
ECC_Addrbus on page 11-293	0x40	32	RW	0x0	MSB bit of address is determined by ADR.
ECC_RData0bus on page 11-293	0x44	32	RO	0x0	Data will be read to this register field.
ECC_RData1bus on page 11-294	0x48	32	RO	0x0	Data will be read to this register field.
ECC_RData2bus on page 11-295	0x4C	32	RO	0x0	Data will be read to this register field.

Register	Offset	Width	Access	Reset Value	Description
ECC_RData3bus on page 11-295	0x50	32	RO	0x0	Data will be read to this register field.
ECC_WData0bus on page 11-296	0x54	32	WO	0x0	Data from the register will be written to the RAM.
ECC_WData1bus on page 11-297	0x58	32	WO	0x0	Data from the register will be written to the RAM.
ECC_WData2bus on page 11-297	0x5C	32	WO	0x0	Data from the register will be written to the RAM.
ECC_WData3bus on page 11-298	0x60	32	WO	0x0	Data from the register will be written to the RAM.
ECC_RDataecc0bus on page 11-299	0x64	32	RO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.
ECC_RDataecc1bus on page 11-299	0x68	32	RO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.
ECC_WDataecc0bus on page 11-300	0x6C	32	WO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.
ECC_WDataecc1bus on page 11-301	0x70	32	WO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Register	Offset	Width	Access	Reset Value	Description
ECC_dbytectrl on page 11-302	0x74	32	RW	0x0	Max number of implemented byte enabled is DAT/8
ECC_accctrl on page 11-303	0x78	32	RW	0x0	These bits determine which byte of data/ecc to write to RAM.
ECC_startacc on page 11-304	0x7C	32	RW	0x0	These bits determine which byte of data/ecc to write to RAM.
ECC_wdctrl on page 11-305	0x80	32	RW	0x0	Bits to Enable/Disable Watch Dog Timer
SERRLKUPA0 on page 11-305	0x90	32	RW	0x0	Single-bit error address in LOOKUP TABLE for PORTA.

dmac_ecc Summary

Base Address: 0xFF8C8000

Register Address Offset	Bit Fields															
ecc_dmac_ecc_register-Block																
IP_REV_ID 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SIREV RO 0x0															
CTRL 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															INITA 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							CNT_RSTA 0x0	Reserved							ECC_EN 0x0

Register Address Offset	Bit Fields															
INITSTAT 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															INITCOMPLETEA 0x0
ERRINTEN 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTEN 0x0
ERRINTENS 0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTS 0x0
ERRINTENR 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTR 0x0
INTMODE 0x1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															INTONCMP 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							INTONOVF 0x0	Reserved							INTMODE 0x0
INTSTAT 0x20	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							DERRPENA 0x0	Reserved							SERRPENA 0x0

Register Address Offset	Bit Fields															
INTTEST 0x24	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							TDERRA 0x0	Reserved							TSERRA 0x0
MODSTAT 0x28	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															CMPFLGA 0x0
DERRADDRA 0x2C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							Address 0x0								
SERRADDRA 0x30	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							Address 0x0								
SERRCNTREG 0x3C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	SERRCNT 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SERRCNT 0x0															
ECC_Addrbus 0x40	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							ECC_AddrBUS 0x0								

Register Address Offset	Bit Fields															
ECC_RData0bus 0x44	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_RData1bus 0x48	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_RData2bus 0x4C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_RData3bus 0x50	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_WData0bus 0x54	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_WData1bus 0x58	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															

Register Address Offset	Bit Fields															
ECC_WData2bus 0x5C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_WData3bus 0x60	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_RDataec0bus 0x64	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved			ECC_RDataecc3BUS 0x0					Reserved			ECC_RDataecc2BUS 0x0				
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved			ECC_RDataecc1BUS 0x0					Reserved			ECC_RDataecc0BUS 0x0				
ECC_RDataec1bus 0x68	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved			ECC_RDataecc7BUS 0x0					Reserved			ECC_RDataecc6BUS 0x0				
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved			ECC_RDataecc5BUS 0x0					Reserved			ECC_RDataecc4BUS 0x0				
ECC_WDataec0bus 0x6C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved			ECC_WDataecc3BUS 0x0					Reserved			ECC_WDataecc2BUS 0x0				
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved			ECC_WDataecc1BUS 0x0					Reserved			ECC_WDataecc0BUS 0x0				
ECC_WDataec1bus 0x70	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved			ECC_WDataecc7BUS 0x0					Reserved			ECC_WDataecc6BUS 0x0				
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved			ECC_WDataecc5BUS 0x0					Reserved			ECC_WDataecc4BUS 0x0				

Register Address Offset	Bit Fields															
ECC_dbytectrl 0x74	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								DBEN 0x0							
ECC_accctrl 0x78	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							RDWR 0x0	Reserved						ECCOVR 0x0	DATAOVR 0x0
ECC_startacc 0x7C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															ENBUSA 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															
ECC_wdctrl 0x80	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															WDEN_RAM 0x0
SERRLKUPA0 0x90	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	VALID RW 0x0	Reserved														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							Address RO 0x0								

IP_REV_ID

Module Instance	Base Address	Register Address
ecc_dmac_ecc_registerBlock	0xFF8C8000	0xFF8C8000

Offset: 0x0

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SIREV RO 0x0															

IP_REV_ID Fields

Bit	Name	Description	Access	Reset
15:0	SIREV	IP Rev # These bits indicate the silicon revision number.	RO	0x0

CTRL

ECC Control Register

Module Instance	Base Address	Register Address
ecc_dmac_ecc_registerBlock	0xFF8C8000	0xFF8C8008

Offset: 0x8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															INITA 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							CNT_RSTA 0x0	Reserved							ECC_EN 0x0

CTRL Fields

Bit	Name	Description	Access	Reset
16	INITA	Enable for the hardware memory initialization PORTA.	RW	0x0
8	CNT_RSTA	Enable to reset internal single-bit error counter A value to zero	RW	0x0
0	ECC_EN	Enable for the ECC detection and correction logic.	RW	0x0

INITSTAT

This bit is used to set the initialize the memory and ecc to a known value

Module Instance	Base Address	Register Address
ecc_dmac_ecc_registerBlock	0xFF8C8000	0xFF8C800C

Offset: 0xC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															INITCOM-PLETEA 0x0

INITSTAT Fields

Bit	Name	Description	Access	Reset
0	INITCOMPLETEA	This bit is used to verify if the hardware memory initialization has completed PORTB.	RW	0x0

ERRINTEN

Error Interrupt enable

Module Instance	Base Address	Register Address
ecc_dmac_ecc_registerBlock	0xFF8C8000	0xFF8C8010

Offset: 0x10

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															SERRINTEN 0x0

ERRINTEN Fields

Bit	Name	Description	Access	Reset
0	SERRINTEN	This bit is used to enable the single bit error interrupt of ECC RAM system	RW	0x0

ERRINTENS

Error Interrupt set

Module Instance	Base Address	Register Address
ecc_dmac_ecc_registerBlock	0xFF8C8000	0xFF8C8014

Offset: 0x14

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															SERRINTS 0x0

ERRINTENS Fields

Bit	Name	Description	Access	Reset
0	SERRINTS	This bit is used to set the single-bit error interrupt bit.	RW	0x0

ERRINTENR

Error Interrupt reset.

Module Instance	Base Address	Register Address
ecc_dmac_ecc_registerBlock	0xFF8C8000	0xFF8C8018

Offset: 0x18

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															SERRINTR 0x0

ERRINTENR Fields

Bit	Name	Description	Access	Reset
0	SERRINTR	This bit is used to reset the single-bit error interrupt bit. Reads reflect SERRINTEN. 1'b0: Writing of zero has no effect. 1'b1: By writing one, this bit will reset SERRINTEN bit to 0. This is performing a bitwise writing of this feature.	RW	0x0

INTMODE

Reads reflect SERRINTEN.

Module Instance	Base Address	Register Address
ecc_dmac_ecc_registerBlock	0xFF8C8000	0xFF8C801C

Offset: 0x1C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															INTONCMP 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							INTONOVF 0x0	Reserved							INTMODE 0x0

INTMODE Fields

Bit	Name	Description	Access	Reset
16	INTONCMP	Enable interrupt on compare.	RW	0x0
8	INTONOVF	Enable interrupt on overflow.	RW	0x0
0	INTMODE	Interrupt mode for single-bit errors.	RW	0x0

INTSTAT

This bit is used to enable interrupt generation on SERR lookup table overflow. When all the entries in the table are valid=1 and this is bit is enabled, serr_req signal will be asserted.

Module Instance	Base Address	Register Address
ecc_dmac_ecc_registerBlock	0xFF8C8000	0xFF8C8020

Offset: 0x20

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							DERRPENA 0x0	Reserved							SERRPENA 0x0

INTSTAT Fields

Bit	Name	Description	Access	Reset
8	DERRPENA	Double-bit error pending for PORTA.	RW	0x0
0	SERRPENA	Single-bit error pending for PORTA.	RW	0x0

INTTEST

This bits is used to test interrupt from ECC RAM to GIC

Module Instance	Base Address	Register Address
ecc_dmac_ecc_registerBlock	0xFF8C8000	0xFF8C8024

Offset: 0x24

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							TDERRA 0x0	Reserved							TSERRA 0x0

INTTEST Fields

Bit	Name	Description	Access	Reset
8	TDERRA	Test PORTA Double-bit error.	RW	0x0
0	TSERRA	Test PORTA Single-bit error.	RW	0x0

MODSTAT

Counter feature status flag

Module Instance	Base Address	Register Address
ecc_dmac_ecc_registerBlock	0xFF8C8000	0xFF8C8028

Offset: 0x28

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															CMPFLGA 0x0

MODSTAT Fields

Bit	Name	Description	Access	Reset
0	CMPFLGA	Port A compare status flag	RW	0x0

DERRADDRA

This register shows the address of PORTA current double-bit error. RAM size will determine the maximum number of address bits.

Module Instance	Base Address	Register Address
ecc_dmac_ecc_registerBlock	0xFF8C8000	0xFF8C802C

Offset: 0x2C

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							Address 0x0								

DERRADDRA Fields

Bit	Name	Description	Access	Reset
8:0	Address	Recent double-bit error address.	RO	0x0

SERRADDRA

This register shows the address of PORTA current single-bit error. RAM size will determine the maximum number of address bits.

Module Instance	Base Address	Register Address
ecc_dmac_ecc_registerBlock	0xFF8C8000	0xFF8C8030

Offset: 0x30

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							Address 0x0								

SERRADDRA Fields

Bit	Name	Description	Access	Reset
8:0	Address	Recent single-bit error address.	RO	0x0

SERRCNTREG

Maximum counter value for single-bit error interrupt

Module Instance	Base Address	Register Address
ecc_dmac_ecc_registerBlock	0xFF8C8000	0xFF8C803C

Offset: 0x3C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SERRCNT 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SERRCNT 0x0															

SERRCNTREG Fields

Bit	Name	Description	Access	Reset
31:0	SERRCNT	Counter value	RW	0x0

ECC_Addrbus

MSB bit of address is determined by ADR.

Module Instance	Base Address	Register Address
ecc_dmac_ecc_registerBlock	0xFF8C8000	0xFF8C8040

Offset: 0x40

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							ECC_AddrBUS 0x0								

ECC_Addrbus Fields

Bit	Name	Description	Access	Reset
8:0	ECC_AddrBUS	Address will be driven to RAM to either read or write the data. Address will be latched by the RAM when the Enbus is asserted.	RW	0x0

ECC_RData0bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_dmac_ecc_registerBlock	0xFF8C8000	0xFF8C8044

Offset: 0x44

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData0bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_RDataBUS	ECC_RDataBUS[31:0].	RO	0x0

ECC_RData1bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_dmac_ecc_registerBlock	0xFF8C8000	0xFF8C8048

Offset: 0x48

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData1bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_RDataBUS	ECC_RDataBUS[63:32].	RO	0x0

ECC_RData2bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_dmac_ecc_registerBlock	0xFF8C8000	0xFF8C804C

Offset: 0x4C

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData2bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_RDataBUS	ECC_RDataBUS[95:64].	RO	0x0

ECC_RData3bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_dmac_ecc_registerBlock	0xFF8C8000	0xFF8C8050

Offset: 0x50

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData3bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_RDataBUS	ECC_RDataBUS[127-96].	RO	0x0

ECC_WData0bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_dmac_ecc_registerBlock	0xFF8C8000	0xFF8C8054

Offset: 0x54

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData0bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_WDataBUS	ECC_WDataBUS[31:0].	WO	0x0

ECC_WData1bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_dmac_ecc_registerBlock	0xFF8C8000	0xFF8C8058

Offset: 0x58

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData1bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_WDataBUS	ECC_WDataBUS[63:32].	WO	0x0

ECC_WData2bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_dmac_ecc_registerBlock	0xFF8C8000	0xFF8C805C

Offset: 0x5C

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData2bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_WDataBUS	ECC_WDataBUS[95-64].	WO	0x0

ECC_WData3bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_dmac_ecc_registerBlock	0xFF8C8000	0xFF8C8060

Offset: 0x60

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData3bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_WDataBUS	ECC_WDataBUS[127-96].	WO	0x0

ECC_RDataecc0bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_dmac_ecc_registerBlock	0xFF8C8000	0xFF8C8064

Offset: 0x64

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved			ECC_RDataecc3BUS 0x0					Reserved			ECC_RDataecc2BUS 0x0				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			ECC_RDataecc1BUS 0x0					Reserved			ECC_RDataecc0BUS 0x0				

ECC_RDataecc0bus Fields

Bit	Name	Description	Access	Reset
28:24	ECC_RDataecc3BUS	Eccdata will be read to this register field.	RO	0x0
20:16	ECC_RDataecc2BUS	Eccdata will be read to this register field.	RO	0x0
12:8	ECC_RDataecc1BUS	Eccdata will be read to this register field.	RO	0x0
4:0	ECC_RDataecc0BUS	Eccdata will be read to this register field.	RO	0x0

ECC_RDataecc1bus

The msb bit for the register is configured based on DAT

parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_dmac_ecc_registerBlock	0xFF8C8000	0xFF8C8068

Offset: 0x68

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved			ECC_RDataecc7BUS 0x0					Reserved			ECC_RDataecc6BUS 0x0				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			ECC_RDataecc5BUS 0x0					Reserved			ECC_RDataecc4BUS 0x0				

ECC_RDataecc1bus Fields

Bit	Name	Description	Access	Reset
28:24	ECC_RDataecc7BUS	Eccdata will be read to this register field.	RO	0x0
20:16	ECC_RDataecc6BUS	Eccdata will be read to this register field.	RO	0x0
12:8	ECC_RDataecc5BUS	Eccdata will be read to this register field.	RO	0x0
4:0	ECC_RDataecc4BUS	Eccdata will be read to this register field.	RO	0x0

ECC_WDataecc0bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_dmac_ecc_registerBlock	0xFF8C8000	0xFF8C806C

Offset: 0x6C

Access: WO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved			ECC_WDataecc3BUS 0x0					Reserved			ECC_WDataecc2BUS 0x0				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			ECC_WDataecc1BUS 0x0					Reserved			ECC_WDataecc0BUS 0x0				

ECC_WDataecc0bus Fields

Bit	Name	Description	Access	Reset
28:24	ECC_WDataecc3BUS	Eccdata from the register will be written to the RAM.	WO	0x0
20:16	ECC_WDataecc2BUS	Eccdata from the register will be written to the RAM.	WO	0x0
12:8	ECC_WDataecc1BUS	Eccdata from the register will be written to the RAM.	WO	0x0
4:0	ECC_WDataecc0BUS	Eccdata from the register will be written to the RAM.	WO	0x0

ECC_WDataecc1bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_dmac_ecc_registerBlock	0xFF8C8000	0xFF8C8070

Offset: 0x70

Access: WO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved			ECC_WDataecc7BUS 0x0					Reserved			ECC_WDataecc6BUS 0x0				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			ECC_WDataecc5BUS 0x0					Reserved			ECC_WDataecc4BUS 0x0				

ECC_WDataecc1bus Fields

Bit	Name	Description	Access	Reset
28:24	ECC_WDataecc7BUS	Eccdata from the register will be written to the RAM.	WO	0x0
20:16	ECC_WDataecc6BUS	Eccdata from the register will be written to the RAM.	WO	0x0
12:8	ECC_WDataecc5BUS	Eccdata from the register will be written to the RAM.	WO	0x0
4:0	ECC_WDataecc4BUS	Eccdata from the register will be written to the RAM.	WO	0x0

ECC_dbytectrl

ECC Data Byte Control. The maximum number of implemented byte enables equals the data width of the RAM divided by 8.

Module Instance	Base Address	Register Address
ecc_dmac_ecc_registerBlock	0xFF8C8000	0xFF8C8074

Offset: 0x74

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								DBEN 0x0							

ECC_dbytectrl Fields

Bit	Name	Description	Access	Reset
7:0	DBEN	Byte or word enable for access.	RW	0x0

ECC_accctrl

These bits determine which byte of data/ecc to write to RAM.

Module Instance	Base Address	Register Address
ecc_dmac_ecc_registerBlock	0xFF8C8000	0xFF8C8078

Offset: 0x78

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							RDWR 0x0	Reserved						ECCOVR 0x0	DATAOVR 0x0

ECC_accctrl Fields

Bit	Name	Description	Access	Reset
8	RDWR	Control for read/write.	RW	0x0
1	ECCOVR	ECC Data Override.	RW	0x0
0	DATAOVR	RAM Data Override. Override the ECC_dataBUS register with RAM data in read mode set by ECC_RW. 1'b0: Data override disabled. 1'b1: Data override enabled.	RW	0x0

ECC_startacc

These bits determine which byte of data/ecc to write to RAM.

Module Instance	Base Address	Register Address
ecc_dmac_ecc_registerBlock	0xFF8C8000	0xFF8C807C

Offset: 0x7C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															ENBUSA 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															

ECC_startacc Fields

Bit	Name	Description	Access	Reset
16	ENBUSA	Start RAM access for PORTA.	RW	0x0

ECC_wdctrl

Bits to Enable/Disable Watch Dog Timer

Module Instance	Base Address	Register Address
ecc_dmac_ecc_registerBlock	0xFF8C8000	0xFF8C8080

Offset: 0x80

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															WDEN_RAM 0x0

ECC_wdctrl Fields

Bit	Name	Description	Access	Reset
0	WDEN_RAM	Enable watchdog timeout for OCP register access to IP RAM.	RW	0x0

SERRLKUPA0

Single-bit error address in LOOKUP TABLE for PORTA.

Module Instance	Base Address	Register Address
ecc_dmac_ecc_registerBlock	0xFF8C8000	0xFF8C8090

Offset: 0x90

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VALID RW 0x0	Reserved														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							Address RO 0x0								

SERRLKUPA0 Fields

Bit	Name	Description	Access	Reset
31	VALID	Valid flag bit. Valid bit indicates if the address in this register is current or stale.	RW	0x0
8:0	Address	Recent Single-bit error address. This register shows the address of the each single-bit error. RAM size will determine the maximum number of address bits. If ram size is 32 Kbytes, bit 30-16 will be reserved and read as zero.	RO	0x0

qspi_ecc Address Map

Module Instance	Base Address	End Address
ecc_qspi_ecc_registerBlock	0xFF8C8400	0xFF8C87FF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
IP_REV_ID on page 11-315	0x0	32	RO	0x0	
CTRL on page 11-316	0x8	32	RW	0x0	ECC Control Register

Register	Offset	Width	Access	Reset Value	Description
INITSTAT on page 11-316	0xC	32	RW	0x0	This bit is used to set the initialize the memory and ecc to a known value
ERRINTEN on page 11-317	0x10	32	RW	0x0	Error Interrupt enable
ERRINTENS on page 11-318	0x14	32	RW	0x0	Error Interrupt set
ERRINTENR on page 11-319	0x18	32	RW	0x0	Error Interrupt reset.
INTMODE on page 11-320	0x1C	32	RW	0x0	Reads reflect SERRINTEN.
INTSTAT on page 11-320	0x20	32	RW	0x0	This bit is used to enable interrupt generation on SERR lookup table overflow. When all the entries in the table are valid=1 and this is bit is enabled, serr_req signal will be asserted.
INTTEST on page 11-321	0x24	32	RW	0x0	This bits is used to test interrupt from ECC RAM to GIC
MODSTAT on page 11-322	0x28	32	RW	0x0	Counter feature status flag
DERRADDRA on page 11-323	0x2C	32	RO	0x0	This register shows the address of PORTA current double-bit error. RAM size will determine the maximum number of address bits.
SERRADDRA on page 11-323	0x30	32	RO	0x0	This register shows the address of PORTA current single-bit error. RAM size will determine the maximum number of address bits.

Register	Offset	Width	Access	Reset Value	Description
SERRCNTREG on page 11-324	0x3C	32	RW	0x0	Maximum counter value for single-bit error interrupt
ECC_Addrbus on page 11-325	0x40	32	RW	0x0	MSB bit of address is determined by ADR.
ECC_RData0bus on page 11-326	0x44	32	RO	0x0	Data will be read to this register field.
ECC_RData1bus on page 11-326	0x48	32	RO	0x0	Data will be read to this register field.
ECC_RData2bus on page 11-327	0x4C	32	RO	0x0	Data will be read to this register field.
ECC_RData3bus on page 11-328	0x50	32	RO	0x0	Data will be read to this register field.
ECC_WData0bus on page 11-328	0x54	32	WO	0x0	Data from the register will be written to the RAM.
ECC_WData1bus on page 11-329	0x58	32	WO	0x0	Data from the register will be written to the RAM.
ECC_WData2bus on page 11-330	0x5C	32	WO	0x0	Data from the register will be written to the RAM.
ECC_WData3bus on page 11-330	0x60	32	WO	0x0	Data from the register will be written to the RAM.
ECC_RDataecc0bus on page 11-331	0x64	32	RO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Register	Offset	Width	Access	Reset Value	Description
ECC_RDataecc1bus on page 11-332	0x68	32	RO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.
ECC_WDataecc0bus on page 11-333	0x6C	32	WO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.
ECC_WDataecc1bus on page 11-334	0x70	32	WO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.
ECC_dbytectrl on page 11-335	0x74	32	RW	0x0	Max number of implemented byte enabled is DAT/8
ECC_accctrl on page 11-336	0x78	32	RW	0x0	These bits determine which byte of data/ecc to write to RAM.
ECC_startacc on page 11-336	0x7C	32	RW	0x0	These bits determine which byte of data/ecc to write to RAM.
ECC_wdctrl on page 11-337	0x80	32	RW	0x0	Bits to Enable/Disable Watch Dog Timer
SERRLKUPA0 on page 11-338	0x90	32	RW	0x0	Single-bit error address in LOOKUP TABLE for PORTA.

qspi_ecc Summary

Base Address: 0xFF8C8400

Register Address Offset	Bit Fields															
ecc_qspi_ecc_register-Block																
IP_REV_ID 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SIREV RO 0x0															
CTRL 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															INITA 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							CNT_RSTA 0x0	Reserved							ECC_EN 0x0
INITSTAT 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															INITCOM-PLETEA 0x0
ERRINTEN 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTEN 0x0
ERRINTENS 0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTS 0x0
ERRINTENR 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTR 0x0

Register Address Offset	Bit Fields															
INTMODE 0x1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															INTONCMP 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							INTONOVF 0x0	Reserved							INTMODE 0x0
INTSTAT 0x20	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							DERRPENA 0x0	Reserved							SERRPENA 0x0
INTTEST 0x24	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							TDERRA 0x0	Reserved							TSERRA 0x0
MODSTAT 0x28	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															CMPFLGA 0x0
DERRADDRA 0x2C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved									Address 0x0						
SERRADDRA 0x30	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved									Address 0x0						

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SERRCNTREG 0x3C	SERRCNT 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SERRCNT 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_Addrbus 0x40	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved									ECC_AddrBUS 0x0						
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RData0bus 0x44	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RData1bus 0x48	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RData2bus 0x4C	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RData3bus 0x50	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															

Register Address Offset	Bit Fields															
ECC_WData0bus 0x54	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_WData1bus 0x58	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_WData2bus 0x5C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_WData3bus 0x60	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_RDataecc0bus 0x64	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_RDataecc3BUS 0x0							Reserved	ECC_RDataecc2BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_RDataecc1BUS 0x0							Reserved	ECC_RDataecc0BUS 0x0						
ECC_RDataecc1bus 0x68	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_RDataecc7BUS 0x0							Reserved	ECC_RDataecc6BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_RDataecc5BUS 0x0							Reserved	ECC_RDataecc4BUS 0x0						

Register Address Offset	Bit Fields															
ECC_WDataecc0bus 0x6C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_WDataecc3BUS 0x0							Reserved	ECC_WDataecc2BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_WDataecc1BUS 0x0							Reserved	ECC_WDataecc0BUS 0x0						
ECC_WDataecc1bus 0x70	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_WDataecc7BUS 0x0							Reserved	ECC_WDataecc6BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_WDataecc5BUS 0x0							Reserved	ECC_WDataecc4BUS 0x0						
ECC_dbytectrl 0x74	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															DBEN 0x0
ECC_accctrl 0x78	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							RDWR 0x0	Reserved						ECCOVR 0x0	DATAOVR 0x0
ECC_startacc 0x7C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															ENBUSA 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															
ECC_wdctrl 0x80	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															WDEN_RAM 0x0

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SERRLKUPA0 0x90	VALID RW 0x0	Reserved														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved									Address RO 0x0						

IP_REV_ID

Module Instance	Base Address	Register Address
ecc_qspi_ecc_registerBlock	0xFF8C8400	0xFF8C8400

Offset: 0x0

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SIREV RO 0x0															

IP_REV_ID Fields

Bit	Name	Description	Access	Reset
15:0	SIREV	IP Rev # These bits indicate the silicon revision number.	RO	0x0

CTRL

ECC Control Register

Module Instance	Base Address	Register Address
ecc_qspi_ecc_registerBlock	0xFF8C8400	0xFF8C8408

Offset: 0x8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															INITA 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							CNT_RSTA 0x0	Reserved							ECC_EN 0x0

CTRL Fields

Bit	Name	Description	Access	Reset
16	INITA	Enable for the hardware memory initialization PORTA.	RW	0x0
8	CNT_RSTA	Enable to reset internal single-bit error counter A value to zero	RW	0x0
0	ECC_EN	Enable for the ECC detection and correction logic.	RW	0x0

INITSTAT

This bit is used to set the initialize the memory and ecc to a known value

Module Instance	Base Address	Register Address
ecc_qspi_ecc_registerBlock	0xFF8C8400	0xFF8C840C

Offset: 0xC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															INITCOM-PLETEA 0x0

INITSTAT Fields

Bit	Name	Description	Access	Reset
0	INITCOMPLETEA	This bit is used to verify if the hardware memory initialization has completed PORTB.	RW	0x0

ERRINTEN

Error Interrupt enable

Module Instance	Base Address	Register Address
ecc_qspi_ecc_registerBlock	0xFF8C8400	0xFF8C8410

Offset: 0x10

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															SERRINTEN 0x0

ERRINTEN Fields

Bit	Name	Description	Access	Reset
0	SERRINTEN	This bit is used to enable the single bit error interrupt of ECC RAM system	RW	0x0

ERRINTENS

Error Interrupt set

Module Instance	Base Address	Register Address
ecc_qspi_ecc_registerBlock	0xFF8C8400	0xFF8C8414

Offset: 0x14

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															SERRINTS 0x0

ERRINTENS Fields

Bit	Name	Description	Access	Reset
0	SERRINTS	This bit is used to set the single-bit error interrupt bit.	RW	0x0

ERRINTENR

Error Interrupt reset.

Module Instance	Base Address	Register Address
ecc_qspi_ecc_registerBlock	0xFF8C8400	0xFF8C8418

Offset: 0x18

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															SERRINTR 0x0

ERRINTENR Fields

Bit	Name	Description	Access	Reset
0	SERRINTR	This bit is used to reset the single-bit error interrupt bit. Reads reflect SERRINTEN. 1'b0: Writing of zero has no effect. 1'b1: By writing one, this bit will reset SERRINTEN bit to 0. This is performing a bitwise writing of this feature.	RW	0x0

INTMODE

Reads reflect SERRINTEN.

Module Instance	Base Address	Register Address
ecc_qspi_ecc_registerBlock	0xFF8C8400	0xFF8C841C

Offset: 0x1C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															INTONCMP 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							INTONOVF 0x0	Reserved							INTMODE 0x0

INTMODE Fields

Bit	Name	Description	Access	Reset
16	INTONCMP	Enable interrupt on compare.	RW	0x0
8	INTONOVF	Enable interrupt on overflow.	RW	0x0
0	INTMODE	Interrupt mode for single-bit errors.	RW	0x0

INTSTAT

This bit is used to enable interrupt generation on SERR lookup table overflow. When all the entries in the table are valid=1 and this is bit is enabled, serr_req signal will be asserted.

Module Instance	Base Address	Register Address
ecc_qspi_ecc_registerBlock	0xFF8C8400	0xFF8C8420

Offset: 0x20

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							DERRPENA 0x0	Reserved							SERRPENA 0x0

INTSTAT Fields

Bit	Name	Description	Access	Reset
8	DERRPENA	Double-bit error pending for PORTA.	RW	0x0
0	SERRPENA	Single-bit error pending for PORTA.	RW	0x0

INTTEST

This bits is used to test interrupt from ECC RAM to GIC

Module Instance	Base Address	Register Address
ecc_qspi_ecc_registerBlock	0xFF8C8400	0xFF8C8424

Offset: 0x24

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							TDERRA 0x0	Reserved							TSERRA 0x0

INTTEST Fields

Bit	Name	Description	Access	Reset
8	TDERRA	Test PORTA Double-bit error.	RW	0x0
0	TSERRA	Test PORTA Single-bit error.	RW	0x0

MODSTAT

Counter feature status flag

Module Instance	Base Address	Register Address
ecc_qspi_ecc_registerBlock	0xFF8C8400	0xFF8C8428

Offset: 0x28

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															CMPFLGA 0x0

MODSTAT Fields

Bit	Name	Description	Access	Reset
0	CMPFLGA	Port A compare status flag	RW	0x0

DERRADDRA

This register shows the address of PORTA current double-bit error. RAM size will determine the maximum number of address bits.

Module Instance	Base Address	Register Address
ecc_qspi_ecc_registerBlock	0xFF8C8400	0xFF8C842C

Offset: 0x2C

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									Address 0x0						

DERRADDRA Fields

Bit	Name	Description	Access	Reset
6:0	Address	Recent double-bit error address.	RO	0x0

SERRADDRA

This register shows the address of PORTA current single-bit error. RAM size will determine the maximum number of address bits.

Module Instance	Base Address	Register Address
ecc_qspi_ecc_registerBlock	0xFF8C8400	0xFF8C8430

Offset: 0x30

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									Address 0x0						

SERRADDRA Fields

Bit	Name	Description	Access	Reset
6:0	Address	Recent single-bit error address.	RO	0x0

SERRCNTREG

Maximum counter value for single-bit error interrupt

Module Instance	Base Address	Register Address
ecc_qspi_ecc_registerBlock	0xFF8C8400	0xFF8C843C

Offset: 0x3C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SERRCNT 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SERRCNT 0x0															

SERRCNTREG Fields

Bit	Name	Description	Access	Reset
31:0	SERRCNT	Counter value	RW	0x0

ECC_Addrbus

MSB bit of address is determined by ADR.

Module Instance	Base Address	Register Address
ecc_qspi_ecc_registerBlock	0xFF8C8400	0xFF8C8440

Offset: 0x40

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									ECC_AddrBUS 0x0						

ECC_Addrbus Fields

Bit	Name	Description	Access	Reset
6:0	ECC_AddrBUS	Address will be driven to RAM to either read or write the data. Address will be latched by the RAM when the Enbus is asserted.	RW	0x0

ECC_RData0bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_qspi_ecc_registerBlock	0xFF8C8400	0xFF8C8444

Offset: 0x44

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData0bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_RDataBUS	ECC_RDataBUS[31:0].	RO	0x0

ECC_RData1bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_qspi_ecc_registerBlock	0xFF8C8400	0xFF8C8448

Offset: 0x48

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData1bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_RDataBUS	ECC_RDataBUS[63:32].	RO	0x0

ECC_RData2bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_qspi_ecc_registerBlock	0xFF8C8400	0xFF8C844C

Offset: 0x4C

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData2bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_RDataBUS	ECC_RDataBUS[95:64].	RO	0x0

ECC_RData3bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_qspi_ecc_registerBlock	0xFF8C8400	0xFF8C8450

Offset: 0x50

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData3bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_RDataBUS	ECC_RDataBUS[127-96].	RO	0x0

ECC_WData0bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_qspi_ecc_registerBlock	0xFF8C8400	0xFF8C8454

Offset: 0x54

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData0bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_WDataBUS	ECC_WDataBUS[31:0].	WO	0x0

ECC_WData1bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_qspi_ecc_registerBlock	0xFF8C8400	0xFF8C8458

Offset: 0x58

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData1bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_WDataBUS	ECC_WDataBUS[63:32].	WO	0x0

ECC_WData2bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_qspi_ecc_registerBlock	0xFF8C8400	0xFF8C845C

Offset: 0x5C

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData2bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_WDataBUS	ECC_WDataBUS[95-64].	WO	0x0

ECC_WData3bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_qspi_ecc_registerBlock	0xFF8C8400	0xFF8C8460

Offset: 0x60

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData3bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_WDataBUS	ECC_WDataBUS[127-96].	WO	0x0

ECC_RDataecc0bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_qspi_ecc_registerBlock	0xFF8C8400	0xFF8C8464

Offset: 0x64

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	ECC_RDataecc3BUS 0x0							Reserved	ECC_RDataecc2BUS 0x0						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	ECC_RDataecc1BUS 0x0							Reserved	ECC_RDataecc0BUS 0x0						

ECC_RDataecc0bus Fields

Bit	Name	Description	Access	Reset
30:24	ECC_RDataecc3BUS	Eccdata will be read to this register field.	RO	0x0
22:16	ECC_RDataecc2BUS	Eccdata will be read to this register field.	RO	0x0
14:8	ECC_RDataecc1BUS	Eccdata will be read to this register field.	RO	0x0
6:0	ECC_RDataecc0BUS	Eccdata will be read to this register field.	RO	0x0

ECC_RDataecc1bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_qspi_ecc_registerBlock	0xFF8C8400	0xFF8C8468

Offset: 0x68

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	ECC_RDataecc7BUS 0x0							Reserved	ECC_RDataecc6BUS 0x0						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	ECC_RDataecc5BUS 0x0							Reserved	ECC_RDataecc4BUS 0x0						

ECC_RDataecc1bus Fields

Bit	Name	Description	Access	Reset
30:24	ECC_RDataecc7BUS	Eccdata will be read to this register field.	RO	0x0
22:16	ECC_RDataecc6BUS	Eccdata will be read to this register field.	RO	0x0
14:8	ECC_RDataecc5BUS	Eccdata will be read to this register field.	RO	0x0
6:0	ECC_RDataecc4BUS	Eccdata will be read to this register field.	RO	0x0

ECC_WDataecc0bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_qspi_ecc_registerBlock	0xFF8C8400	0xFF8C846C

Offset: 0x6C

Access: WO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	ECC_WDataecc3BUS 0x0							Reserved	ECC_WDataecc2BUS 0x0						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	ECC_WDataecc1BUS 0x0							Reserved	ECC_WDataecc0BUS 0x0						

ECC_WDataecc0bus Fields

Bit	Name	Description	Access	Reset
30:24	ECC_WDataecc3BUS	Eccdata from the register will be written to the RAM.	WO	0x0
22:16	ECC_WDataecc2BUS	Eccdata from the register will be written to the RAM.	WO	0x0
14:8	ECC_WDataecc1BUS	Eccdata from the register will be written to the RAM.	WO	0x0
6:0	ECC_WDataecc0BUS	Eccdata from the register will be written to the RAM.	WO	0x0

ECC_WDataecc1bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_qspi_ecc_registerBlock	0xFF8C8400	0xFF8C8470

Offset: 0x70

Access: WO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	ECC_WDataecc7BUS 0x0							Reserved	ECC_WDataecc6BUS 0x0						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	ECC_WDataecc5BUS 0x0							Reserved	ECC_WDataecc4BUS 0x0						

ECC_WDataecc1bus Fields

Bit	Name	Description	Access	Reset
30:24	ECC_WDataecc7BUS	Eccdata from the register will be written to the RAM.	WO	0x0
22:16	ECC_WDataecc6BUS	Eccdata from the register will be written to the RAM.	WO	0x0
14:8	ECC_WDataecc5BUS	Eccdata from the register will be written to the RAM.	WO	0x0
6:0	ECC_WDataecc4BUS	Eccdata from the register will be written to the RAM.	WO	0x0

ECC_dbytectrl

ECC Data Byte Control. The maximum number of implemented byte enables equals the data width of the RAM divided by 8.

Module Instance	Base Address	Register Address
ecc_qspi_ecc_registerBlock	0xFF8C8400	0xFF8C8474

Offset: 0x74

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															DBEN 0x0

ECC_dbytectrl Fields

Bit	Name	Description	Access	Reset
0	DBEN	Byte or word enable for access.	RW	0x0

ECC_accctrl

These bits determine which byte of data/ecc to write to RAM.

Module Instance	Base Address	Register Address
ecc_qspi_ecc_registerBlock	0xFF8C8400	0xFF8C8478

Offset: 0x78

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							RDWR 0x0	Reserved						ECCOVR 0x0	DATAOVR 0x0

ECC_accctrl Fields

Bit	Name	Description	Access	Reset
8	RDWR	Control for read/write.	RW	0x0
1	ECCOVR	ECC Data Override.	RW	0x0
0	DATAOVR	RAM Data Override. Override the ECC_dataBUS register with RAM data in read mode set by ECC_RW. 1'b0: Data override disabled. 1'b1: Data override enabled.	RW	0x0

ECC_startacc

These bits determine which byte of data/ecc to write to RAM.

Module Instance	Base Address	Register Address
ecc_qspi_ecc_registerBlock	0xFF8C8400	0xFF8C847C

Offset: 0x7C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															ENBUSA 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															

ECC_startacc Fields

Bit	Name	Description	Access	Reset
16	ENBUSA	Start RAM access for PORTA.	RW	0x0

ECC_wdctrl

Bits to Enable/Disable Watch Dog Timer

Module Instance	Base Address	Register Address
ecc_qspi_ecc_registerBlock	0xFF8C8400	0xFF8C8480

Offset: 0x80

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															WDEN_RAM 0x0

ECC_wdctrl Fields

Bit	Name	Description	Access	Reset
0	WDEN_RAM	Enable watchdog timeout for OCP register access to IP RAM.	RW	0x0

SERRLKUPA0

Single-bit error address in LOOKUP TABLE for PORTA.

Module Instance	Base Address	Register Address
ecc_qspi_ecc_registerBlock	0xFF8C8400	0xFF8C8490

Offset: 0x90

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VALID RW 0x0	Reserved														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									Address RO 0x0						

SERRLKUPA0 Fields

Bit	Name	Description	Access	Reset
31	VALID	Valid flag bit. Valid bit indicates if the address in this register is current or stale.	RW	0x0
6:0	Address	Recent Single-bit error address. This register shows the address of the each single-bit error. RAM size will determine the maximum number of address bits. If ram size is 32 Kbytes, bit 30-16 will be reserved and read as zero.	RO	0x0

usb_ecc Address Map

Module Instance	Base Address	End Address
ecc_otg0_ecc_registerBlock	0xFF8C8800	0xFF8C8BFF
ecc_otg1_ecc_registerBlock	0xFF8C8C00	0xFF9FFFFF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
IP_REV_ID on page 11-353	0x0	32	RO	0x0	
CTRL on page 11-354	0x8	32	RW	0x0	ECC Control Register
INITSTAT on page 11-354	0xC	32	RW	0x0	This bit is used to set the initialize the memory and ecc to a known value
ERRINTEN on page 11-355	0x10	32	RW	0x0	Error Interrupt enable
ERRINTENS on page 11-356	0x14	32	RW	0x0	Error Interrupt set
ERRINTENR on page 11-357	0x18	32	RW	0x0	Error Interrupt reset.

Register	Offset	Width	Access	Reset Value	Description
INTMODE on page 11-358	0x1C	32	RW	0x0	Reads reflect SERRINTEN.
INTSTAT on page 11-358	0x20	32	RW	0x0	This bit is used to enable interrupt generation on SERR lookup table overflow. When all the entries in the table are valid=1 and this is bit is enabled, serr_req signal will be asserted.
INTTEST on page 11-359	0x24	32	RW	0x0	This bits is used to test interrupt from ECC RAM to GIC
MODSTAT on page 11-360	0x28	32	RW	0x0	Counter feature status flag
DERRADDRA on page 11-361	0x2C	32	RO	0x0	This register shows the address of PORTA current double-bit error. RAM size will determine the maximum number of address bits.
SERRADDRA on page 11-361	0x30	32	RO	0x0	This register shows the address of PORTA current single-bit error. RAM size will determine the maximum number of address bits.
SERRCNTREG on page 11-362	0x3C	32	RW	0x0	Maximum counter value for single-bit error interrupt
ECC_Addrbus on page 11-363	0x40	32	RW	0x0	MSB bit of address is determined by ADR.
ECC_RData0bus on page 11-364	0x44	32	RO	0x0	Data will be read to this register field.
ECC_RData1bus on page 11-364	0x48	32	RO	0x0	Data will be read to this register field.
ECC_RData2bus on page 11-365	0x4C	32	RO	0x0	Data will be read to this register field.

Register	Offset	Width	Access	Reset Value	Description
ECC_RData3bus on page 11-366	0x50	32	RO	0x0	Data will be read to this register field.
ECC_WData0bus on page 11-367	0x54	32	WO	0x0	Data from the register will be written to the RAM.
ECC_WData1bus on page 11-367	0x58	32	WO	0x0	Data from the register will be written to the RAM.
ECC_WData2bus on page 11-368	0x5C	32	WO	0x0	Data from the register will be written to the RAM.
ECC_WData3bus on page 11-369	0x60	32	WO	0x0	Data from the register will be written to the RAM.
ECC_RDataecc0bus on page 11-369	0x64	32	RO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.
ECC_RDataecc1bus on page 11-370	0x68	32	RO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.
ECC_WDataecc0bus on page 11-371	0x6C	32	WO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.
ECC_WDataecc1bus on page 11-372	0x70	32	WO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Register	Offset	Width	Access	Reset Value	Description
ECC_dbytectrl on page 11-373	0x74	32	RW	0x0	Max number of implemented byte enabled is DAT/8
ECC_accctrl on page 11-374	0x78	32	RW	0x0	These bits determine which byte of data/ecc to write to RAM.
ECC_startacc on page 11-375	0x7C	32	RW	0x0	These bits determine which byte of data/ecc to write to RAM.
ECC_wdctrl on page 11-376	0x80	32	RW	0x0	Bits to Enable/Disable Watch Dog Timer
SERRLKUPA0 on page 11-376	0x90	32	RW	0x0	Single-bit error address in LOOKUP TABLE for PORTA.

usb_ecc Summary

Module Instance	Base Address
ecc_otg0_ecc_registerBlock	0xFF8C8800
ecc_otg1_ecc_registerBlock	0xFF8C8C00

Register Address Offset	Bit Fields															
ecc_otg0_ecc_register-Block																
IP_REV_ID 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SIREV RO 0x0															

Register Address Offset	Bit Fields															
CTRL 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															INITA 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							CNT_RSTA 0x0	Reserved							ECC_EN 0x0
INITSTAT 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															INITCOM-PLETEA 0x0
ERRINTEN 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTEN 0x0
ERRINTENS 0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTS 0x0
ERRINTENR 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTR 0x0
INTMODE 0x1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															INTONCMP 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							INTONOVF 0x0	Reserved							INTMODE 0x0

Register Address Offset	Bit Fields															
INTSTAT 0x20	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							DERRPENA 0x0	Reserved							SERRPENA 0x0
INTTEST 0x24	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							TDERRA 0x0	Reserved							TSERRA 0x0
MODSTAT 0x28	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															CMPFLGA 0x0
DERRADDRA 0x2C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved			Address 0x0												
SERRADDRA 0x30	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved			Address 0x0												
SERRCNTREG 0x3C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	SERRCNT 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SERRCNT 0x0															

Register Address Offset	Bit Fields															
ECC_Addrbus 0x40	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved			ECC_AddrBUS 0x0												
ECC_RData0bus 0x44	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_RData1bus 0x48	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													ECC_RDataBUS 0x0		
ECC_RData2bus 0x4C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_RData3bus 0x50	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_WData0bus 0x54	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															

Register Address Offset	Bit Fields															
ECC_WData1bus 0x58	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													ECC_WDataBUS 0x0		
ECC_WData2bus 0x5C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_WData3bus 0x60	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_RDataecc0bus 0x64	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_RDataecc3BUS 0x0							Reserved	ECC_RDataecc2BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_RDataecc1BUS 0x0							Reserved	ECC_RDataecc0BUS 0x0						
ECC_RDataecc1bus 0x68	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_RDataecc7BUS 0x0							Reserved	ECC_RDataecc6BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_RDataecc5BUS 0x0							Reserved	ECC_RDataecc4BUS 0x0						
ECC_WDataecc0bus 0x6C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_WDataecc3BUS 0x0							Reserved	ECC_WDataecc2BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_WDataecc1BUS 0x0							Reserved	ECC_WDataecc0BUS 0x0						

Register Address Offset	Bit Fields															
ECC_WDataecc1bus 0x70	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_WDataecc7BUS 0x0							Reserved	ECC_WDataecc6BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_WDataecc5BUS 0x0							Reserved	ECC_WDataecc4BUS 0x0						
ECC_dbytectrl 0x74	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															DBEN 0x0
ECC_accctrl 0x78	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							RDWR 0x0	Reserved						ECCOVR 0x0	DATAOVR 0x0
ECC_startacc 0x7C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															ENBUSA 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															
ECC_wdctrl 0x80	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															WDEN_RAM 0x0
SERRLKUPA0 0x90	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	VALID RW 0x0	Reserved														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved			Address RO 0x0												
ecc_otg1_ecc_register-Block																

Register Address Offset	Bit Fields															
IP_REV_ID 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SIREV RO 0x0															
CTRL 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															INITA 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							CNT_RSTA 0x0	Reserved							ECC_EN 0x0
INITSTAT 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															INITCOM-PLETEA 0x0
ERRINTEN 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTEN 0x0
ERRINTENS 0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTS 0x0
ERRINTENR 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTR 0x0

Register Address Offset	Bit Fields															
INTMODE 0x1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															INTONCMP 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							INTONOVF 0x0	Reserved							INTMODE 0x0
INTSTAT 0x20	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							DERRPENA 0x0	Reserved							SERRPENA 0x0
INTTEST 0x24	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							TDERRA 0x0	Reserved							TSERRA 0x0
MODSTAT 0x28	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															CMPFLGA 0x0
DERRADDRA 0x2C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved			Address 0x0												
SERRADDRA 0x30	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved			Address 0x0												

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SERRCNTREG 0x3C	SERRCNT 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SERRCNT 0x0															
ECC_Addrbus 0x40	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved			ECC_AddrBUS 0x0												
ECC_RData0bus 0x44	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_RData1bus 0x48	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													ECC_RDataBUS 0x0		
ECC_RData2bus 0x4C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_RData3bus 0x50	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															

Register Address Offset	Bit Fields															
ECC_WData0bus 0x54	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_WData1bus 0x58	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													ECC_WDataBUS 0x0		
ECC_WData2bus 0x5C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_WData3bus 0x60	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_RDataecc0bus 0x64	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_RDataecc3BUS 0x0							Reserved	ECC_RDataecc2BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_RDataecc1BUS 0x0							Reserved	ECC_RDataecc0BUS 0x0						
ECC_RDataecc1bus 0x68	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_RDataecc7BUS 0x0							Reserved	ECC_RDataecc6BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_RDataecc5BUS 0x0							Reserved	ECC_RDataecc4BUS 0x0						

Register Address Offset	Bit Fields															
ECC_WDataecc0bus 0x6C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_WDataecc3BUS 0x0							Reserved	ECC_WDataecc2BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_WDataecc1BUS 0x0							Reserved	ECC_WDataecc0BUS 0x0						
ECC_WDataecc1bus 0x70	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_WDataecc7BUS 0x0							Reserved	ECC_WDataecc6BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_WDataecc5BUS 0x0							Reserved	ECC_WDataecc4BUS 0x0						
ECC_dbytectrl 0x74	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															DBEN 0x0
ECC_accctrl 0x78	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							RDWR 0x0	Reserved						ECCOVR 0x0	DATAOVR 0x0
ECC_startacc 0x7C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															ENBUSA 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															
ECC_wdctrl 0x80	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															WDEN_RAM 0x0

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SERRLKUPA0 0x90	VALID RW 0x0	Reserved														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved			Address RO 0x0												

IP_REV_ID

Module Instance	Base Address	Register Address
ecc_otg0_ecc_registerBlock	0xFF8C8800	0xFF8C8800
ecc_otg1_ecc_registerBlock	0xFF8C8C00	0xFF8C8C00

Offset: 0x0

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SIREV RO 0x0															

IP_REV_ID Fields

Bit	Name	Description	Access	Reset
15:0	SIREV	IP Rev # These bits indicate the silicon revision number.	RO	0x0

CTRL

ECC Control Register

Module Instance	Base Address	Register Address
ecc_otg0_ecc_registerBlock	0xFF8C8800	0xFF8C8808
ecc_otg1_ecc_registerBlock	0xFF8C8C00	0xFF8C8C08

Offset: 0x8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															INITA 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							CNT_RSTA 0x0	Reserved							ECC_EN 0x0

CTRL Fields

Bit	Name	Description	Access	Reset
16	INITA	Enable for the hardware memory initialization PORTA.	RW	0x0
8	CNT_RSTA	Enable to reset internal single-bit error counter A value to zero	RW	0x0
0	ECC_EN	Enable for the ECC detection and correction logic.	RW	0x0

INITSTAT

This bit is used to set the initialize the memory and ecc to a known value

Module Instance	Base Address	Register Address
ecc_otg0_ecc_registerBlock	0xFF8C8800	0xFF8C880C
ecc_otg1_ecc_registerBlock	0xFF8C8C00	0xFF8C8C0C

Offset: 0xC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															INITCOMPLETEA 0x0

INITSTAT Fields

Bit	Name	Description	Access	Reset
0	INITCOMPLETEA	This bit is used to verify if the hardware memory initialization has completed PORTB.	RW	0x0

ERRINTEN

Error Interrupt enable

Module Instance	Base Address	Register Address
ecc_otg0_ecc_registerBlock	0xFF8C8800	0xFF8C8810
ecc_otg1_ecc_registerBlock	0xFF8C8C00	0xFF8C8C10

Offset: 0x10

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															SERRINTEN 0x0

ERRINTEN Fields

Bit	Name	Description	Access	Reset
0	SERRINTEN	This bit is used to enable the single bit error interrupt of ECC RAM system	RW	0x0

ERRINTENS

Error Interrupt set

Module Instance	Base Address	Register Address
ecc_otg0_ecc_registerBlock	0xFF8C8800	0xFF8C8814
ecc_otg1_ecc_registerBlock	0xFF8C8C00	0xFF8C8C14

Offset: 0x14

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															SERRINTS 0x0

ERRINTENS Fields

Bit	Name	Description	Access	Reset
0	SERRINTS	This bit is used to set the single-bit error interrupt bit.	RW	0x0

ERRINTENR

Error Interrupt reset.

Module Instance	Base Address	Register Address
ecc_otg0_ecc_registerBlock	0xFF8C8800	0xFF8C8818
ecc_otg1_ecc_registerBlock	0xFF8C8C00	0xFF8C8C18

Offset: 0x18

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															SERRINTR 0x0

ERRINTENR Fields

Bit	Name	Description	Access	Reset
0	SERRINTR	This bit is used to reset the single-bit error interrupt bit. Reads reflect SERRINTEN. 1'b0: Writing of zero has no effect. 1'b1: By writing one, this bit will reset SERRINTEN bit to 0. This is performing a bitwise writing of this feature.	RW	0x0

INTMODE

Reads reflect SERRINTEN.

Module Instance	Base Address	Register Address
ecc_otg0_ecc_registerBlock	0xFF8C8800	0xFF8C881C
ecc_otg1_ecc_registerBlock	0xFF8C8C00	0xFF8C8C1C

Offset: 0x1C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															INTONCMP 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							INTONOVF 0x0	Reserved							INTMODE 0x0

INTMODE Fields

Bit	Name	Description	Access	Reset
16	INTONCMP	Enable interrupt on compare.	RW	0x0
8	INTONOVF	Enable interrupt on overflow.	RW	0x0
0	INTMODE	Interrupt mode for single-bit errors.	RW	0x0

INTSTAT

This bit is used to enable interrupt generation on SERR lookup table overflow. When all the entries in the table are valid=1 and this is bit is enabled, serr_req signal will be asserted.

Module Instance	Base Address	Register Address
ecc_otg0_ecc_registerBlock	0xFF8C8800	0xFF8C8820
ecc_otg1_ecc_registerBlock	0xFF8C8C00	0xFF8C8C20

Offset: 0x20

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							DERRPENA 0x0	Reserved							SERRPENA 0x0

INTSTAT Fields

Bit	Name	Description	Access	Reset
8	DERRPENA	Double-bit error pending for PORTA.	RW	0x0
0	SERRPENA	Single-bit error pending for PORTA.	RW	0x0

INTTEST

This bits is used to test interrupt from ECC RAM to GIC

Module Instance	Base Address	Register Address
ecc_otg0_ecc_registerBlock	0xFF8C8800	0xFF8C8824
ecc_otg1_ecc_registerBlock	0xFF8C8C00	0xFF8C8C24

Offset: 0x24

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							TDERRA 0x0	Reserved							TSERRA 0x0

INTTEST Fields

Bit	Name	Description	Access	Reset
8	TDERRA	Test PORTA Double-bit error.	RW	0x0
0	TSERRA	Test PORTA Single-bit error.	RW	0x0

MODSTAT

Counter feature status flag

Module Instance	Base Address	Register Address
ecc_otg0_ecc_registerBlock	0xFF8C8800	0xFF8C8828
ecc_otg1_ecc_registerBlock	0xFF8C8C00	0xFF8C8C28

Offset: 0x28

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															CMPFLGA 0x0

MODSTAT Fields

Bit	Name	Description	Access	Reset
0	CMPFLGA	Port A compare status flag	RW	0x0

DERRADDRA

This register shows the address of PORTA current double-bit error. RAM size will determine the maximum number of address bits.

Module Instance	Base Address	Register Address
ecc_otg0_ecc_registerBlock	0xFF8C8800	0xFF8C882C
ecc_otg1_ecc_registerBlock	0xFF8C8C00	0xFF8C8C2C

Offset: 0x2C

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			Address 0x0												

DERRADDRA Fields

Bit	Name	Description	Access	Reset
12:0	Address	Recent double-bit error address.	RO	0x0

SERRADDRA

This register shows the address of PORTA current single-bit error. RAM size will determine the maximum number of address bits.

Module Instance	Base Address	Register Address
ecc_otg0_ecc_registerBlock	0xFF8C8800	0xFF8C8830
ecc_otg1_ecc_registerBlock	0xFF8C8C00	0xFF8C8C30

Offset: 0x30

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			Address 0x0												

SERRADDRA Fields

Bit	Name	Description	Access	Reset
12:0	Address	Recent single-bit error address.	RO	0x0

SERRCNTREG

Maximum counter value for single-bit error interrupt

Module Instance	Base Address	Register Address
ecc_otg0_ecc_registerBlock	0xFF8C8800	0xFF8C883C
ecc_otg1_ecc_registerBlock	0xFF8C8C00	0xFF8C8C3C

Offset: 0x3C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SERRCNT 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SERRCNT 0x0															

SERRCNTREG Fields

Bit	Name	Description	Access	Reset
31:0	SERRCNT	Counter value	RW	0x0

ECC_Addrbus

MSB bit of address is determined by ADR.

Module Instance	Base Address	Register Address
ecc_otg0_ecc_registerBlock	0xFF8C8800	0xFF8C8840
ecc_otg1_ecc_registerBlock	0xFF8C8C00	0xFF8C8C40

Offset: 0x40

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			ECC_AddrBUS 0x0												

ECC_Addrbus Fields

Bit	Name	Description	Access	Reset
12:0	ECC_AddrBUS	Address will be driven to RAM to either read or write the data. Address will be latched by the RAM when the Enbus is asserted.	RW	0x0

ECC_RData0bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_otg0_ecc_registerBlock	0xFF8C8800	0xFF8C8844
ecc_otg1_ecc_registerBlock	0xFF8C8C00	0xFF8C8C44

Offset: 0x44

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData0bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_RDataBUS	ECC_RDataBUS[31:0].	RO	0x0

ECC_RData1bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_otg0_ecc_registerBlock	0xFF8C8800	0xFF8C8848
ecc_otg1_ecc_registerBlock	0xFF8C8C00	0xFF8C8C48

Offset: 0x48

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													ECC_RDataBUS		
													0x0		

ECC_RData1bus Fields

Bit	Name	Description	Access	Reset
2:0	ECC_RDataBUS	ECC_RDataBUS[63:32].	RO	0x0

ECC_RData2bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_otg0_ecc_registerBlock	0xFF8C8800	0xFF8C884C
ecc_otg1_ecc_registerBlock	0xFF8C8C00	0xFF8C8C4C

Offset: 0x4C

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData2bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_RDataBUS	ECC_RDataBUS[95:64].	RO	0x0

ECC_RData3bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_otg0_ecc_registerBlock	0xFF8C8800	0xFF8C8850
ecc_otg1_ecc_registerBlock	0xFF8C8C00	0xFF8C8C50

Offset: 0x50

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData3bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_RDataBUS	ECC_RDataBUS[127-96].	RO	0x0

ECC_WData0bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_otg0_ecc_registerBlock	0xFF8C8800	0xFF8C8854
ecc_otg1_ecc_registerBlock	0xFF8C8C00	0xFF8C8C54

Offset: 0x54

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData0bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_WDataBUS	ECC_WDataBUS[31:0].	WO	0x0

ECC_WData1bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_otg0_ecc_registerBlock	0xFF8C8800	0xFF8C8858
ecc_otg1_ecc_registerBlock	0xFF8C8C00	0xFF8C8C58

Offset: 0x58

Access: WO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													ECC_WDataBUS 0x0		

ECC_WData1bus Fields

Bit	Name	Description	Access	Reset
2:0	ECC_WDataBUS	ECC_WDataBUS[63:32].	WO	0x0

ECC_WData2bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_otg0_ecc_registerBlock	0xFF8C8800	0xFF8C885C
ecc_otg1_ecc_registerBlock	0xFF8C8C00	0xFF8C8C5C

Offset: 0x5C

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData2bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_WDataBUS	ECC_WDataBUS[95-64].	WO	0x0

ECC_WData3bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_otg0_ecc_registerBlock	0xFF8C8800	0xFF8C8860
ecc_otg1_ecc_registerBlock	0xFF8C8C00	0xFF8C8C60

Offset: 0x60

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData3bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_WDataBUS	ECC_WDataBUS[127-96].	WO	0x0

ECC_RDataecc0bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_otg0_ecc_registerBlock	0xFF8C8800	0xFF8C8864
ecc_otg1_ecc_registerBlock	0xFF8C8C00	0xFF8C8C64

Offset: 0x64

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	ECC_RDataecc3BUS 0x0							Reserved	ECC_RDataecc2BUS 0x0						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	ECC_RDataecc1BUS 0x0							Reserved	ECC_RDataecc0BUS 0x0						

ECC_RDataecc0bus Fields

Bit	Name	Description	Access	Reset
30:24	ECC_RDataecc3BUS	Eccdata will be read to this register field.	RO	0x0
22:16	ECC_RDataecc2BUS	Eccdata will be read to this register field.	RO	0x0
14:8	ECC_RDataecc1BUS	Eccdata will be read to this register field.	RO	0x0
6:0	ECC_RDataecc0BUS	Eccdata will be read to this register field.	RO	0x0

ECC_RDataecc1bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_otg0_ecc_registerBlock	0xFF8C8800	0xFF8C8868
ecc_otg1_ecc_registerBlock	0xFF8C8C00	0xFF8C8C68

Offset: 0x68

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields																
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	ECC_RDataecc7BUS 0x0							Reserved	ECC_RDataecc6BUS 0x0						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	ECC_RDataecc5BUS 0x0							Reserved	ECC_RDataecc4BUS 0x0						

ECC_RDataecc1bus Fields

Bit	Name	Description	Access	Reset
30:24	ECC_RDataecc7BUS	Eccdata will be read to this register field.	RO	0x0
22:16	ECC_RDataecc6BUS	Eccdata will be read to this register field.	RO	0x0
14:8	ECC_RDataecc5BUS	Eccdata will be read to this register field.	RO	0x0
6:0	ECC_RDataecc4BUS	Eccdata will be read to this register field.	RO	0x0

ECC_WDataecc0bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_otg0_ecc_registerBlock	0xFF8C8800	0xFF8C886C
ecc_otg1_ecc_registerBlock	0xFF8C8C00	0xFF8C8C6C

Offset: 0x6C

Access: WO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields																
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	ECC_WDataecc3BUS 0x0							Reserved	ECC_WDataecc2BUS 0x0						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	ECC_WDataecc1BUS 0x0							Reserved	ECC_WDataecc0BUS 0x0						

ECC_WDataecc0bus Fields

Bit	Name	Description	Access	Reset
30:24	ECC_WDataecc3BUS	Eccdata from the register will be written to the RAM.	WO	0x0
22:16	ECC_WDataecc2BUS	Eccdata from the register will be written to the RAM.	WO	0x0
14:8	ECC_WDataecc1BUS	Eccdata from the register will be written to the RAM.	WO	0x0
6:0	ECC_WDataecc0BUS	Eccdata from the register will be written to the RAM.	WO	0x0

ECC_WDataecc1bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_otg0_ecc_registerBlock	0xFF8C8800	0xFF8C8870
ecc_otg1_ecc_registerBlock	0xFF8C8C00	0xFF8C8C70

Offset: 0x70

Access: WO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields																
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	ECC_WDataecc7BUS 0x0							Reserved	ECC_WDataecc6BUS 0x0						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	ECC_WDataecc5BUS 0x0							Reserved	ECC_WDataecc4BUS 0x0						

ECC_WDataecc1bus Fields

Bit	Name	Description	Access	Reset
30:24	ECC_WDataecc7BUS	Eccdata from the register will be written to the RAM.	WO	0x0
22:16	ECC_WDataecc6BUS	Eccdata from the register will be written to the RAM.	WO	0x0
14:8	ECC_WDataecc5BUS	Eccdata from the register will be written to the RAM.	WO	0x0
6:0	ECC_WDataecc4BUS	Eccdata from the register will be written to the RAM.	WO	0x0

ECC_dbytectrl

ECC Data Byte Control. The maximum number of implemented byte enables equals the data width of the RAM divided by 8.

Module Instance	Base Address	Register Address
ecc_otg0_ecc_registerBlock	0xFF8C8800	0xFF8C8874
ecc_otg1_ecc_registerBlock	0xFF8C8C00	0xFF8C8C74

Offset: 0x74

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															DBEN 0x0

ECC_dbytectrl Fields

Bit	Name	Description	Access	Reset
0	DBEN	Byte or word enable for access.	RW	0x0

ECC_accctrl

These bits determine which byte of data/ecc to write to RAM.

Module Instance	Base Address	Register Address
ecc_otg0_ecc_registerBlock	0xFF8C8800	0xFF8C8878
ecc_otg1_ecc_registerBlock	0xFF8C8C00	0xFF8C8C78

Offset: 0x78

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							RDWR 0x0	Reserved						ECCOVR 0x0	DATAOVR 0x0

ECC_accctrl Fields

Bit	Name	Description	Access	Reset
8	RDWR	Control for read/write.	RW	0x0
1	ECCOVR	ECC Data Override.	RW	0x0
0	DATAOVR	RAM Data Override. Override the ECC_dataBUS register with RAM data in read mode set by ECC_RW. 1'b0: Data override disabled. 1'b1: Data override enabled.	RW	0x0

ECC_startacc

These bits determine which byte of data/ecc to write to RAM.

Module Instance	Base Address	Register Address
ecc_otg0_ecc_registerBlock	0xFF8C8800	0xFF8C887C
ecc_otg1_ecc_registerBlock	0xFF8C8C00	0xFF8C8C7C

Offset: 0x7C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															ENBUSA 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															

ECC_startacc Fields

Bit	Name	Description	Access	Reset
16	ENBUSA	Start RAM access for PORTA.	RW	0x0

ECC_wdctrl

Bits to Enable/Disable Watch Dog Timer

Module Instance	Base Address	Register Address
ecc_otg0_ecc_registerBlock	0xFF8C8800	0xFF8C8880
ecc_otg1_ecc_registerBlock	0xFF8C8C00	0xFF8C8C80

Offset: 0x80

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															WDEN_RAM 0x0

ECC_wdctrl Fields

Bit	Name	Description	Access	Reset
0	WDEN_RAM	Enable watchdog timeout for OCP register access to IP RAM.	RW	0x0

SERRLKUPA0

Single-bit error address in LOOKUP TABLE for PORTA.

Module Instance	Base Address	Register Address
ecc_otg0_ecc_registerBlock	0xFF8C8800	0xFF8C8890
ecc_otg1_ecc_registerBlock	0xFF8C8C00	0xFF8C8C90

Offset: 0x90

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VALID RW 0x0	Reserved														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			Address RO 0x0												

SERRLKUPA0 Fields

Bit	Name	Description	Access	Reset
31	VALID	Valid flag bit. Valid bit indicates if the address in this register is current or stale.	RW	0x0
12:0	Address	Recent Single-bit error address. This register shows the address of the each single-bit error. RAM size will determine the maximum number of address bits. If ram size is 32 Kbytes, bit 30-16 will be reserved and read as zero.	RO	0x0

Revision History

Date	Version	Changes
October 2016	2016.10.28	Maintenance Release
May 2016	2016.05.27	Maintenance Release
May 2016	2016.05.03	Maintenance Release
November 2015	2015.11.02	• Added "ECC Bits Required Based on Data Width" table to "Error Checking and Correction Algorithm" section • Added 136-bit Hamming Matrix figure to "Error Checking and Correction Algorithm" section

Date	Version	Changes
May 2015	2015.05.04	• Added 35-bit Hamming Matrix figure to "Error Checking and Correction Algorithm" section • Added "ECC Controller Address Map and Register Definitions" section
December 2014	2014.12.15	Added the "RAM and ECC Memory Organization Example" subsection to the "ECC Structure" section Added the following subsections in the "Indirect Memory Access" section: • "Watchdog Timer" • "Data Correction" • "Error Injection" • "Memory Testing" • "Error Checking and Correction Algorithm"
August 2014	2014.08.18	Initial Release

On-Chip Memory 12

2016.10.28

a10_5v4 Subscribe Send Feedback

The hard processor system (HPS) contains two types of on-chip memory. The on-chip memory types are:

- On-chip RAM—The on-chip RAM provides 256 KB of general-purpose single-port RAM
- Boot ROM—The boot ROM provides 128 KB and is available to store the code required to boot the HPS from cold or warm reset

Both on-chip memories connect to the level 3 (L3) interconnect.

Related Information

- **On-Chip RAM** on page 12-1
- **Boot ROM** on page 12-4

On-Chip RAM

Features of the On-Chip RAM

The on-chip RAM offers the following features:

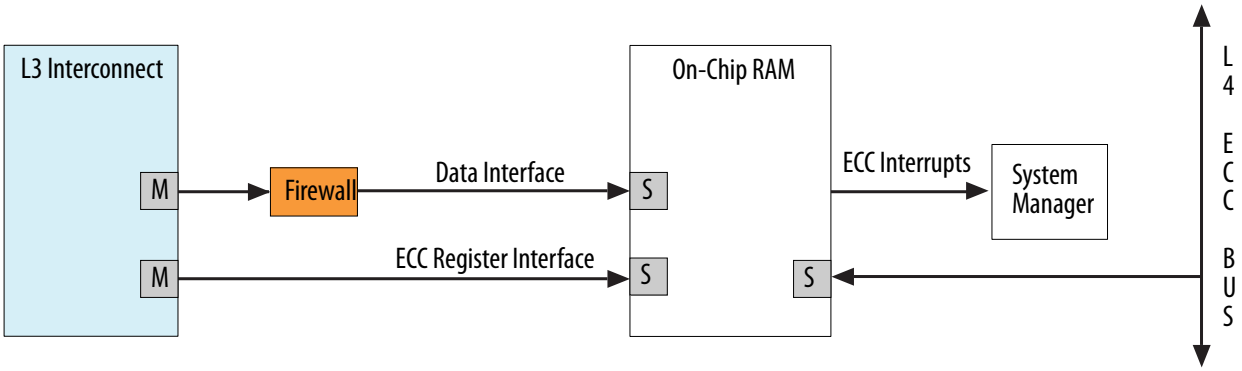
- 64-bit slave interface
- 256 KB of single-ported RAM
- Error correction code (ECC) support
- Capable of data transfers for every clock cycle
- Security firewall
- Supports scrambling and reset of memory on detection of a tamper event

Related Information

Clock Manager on page 2-1

On-Chip RAM Block Diagram and System Integration

Figure 12-1: On-Chip RAM Block Diagram



The on-chip RAM and L3 interconnect transfers data through a 64-bit interface that passes through a firewall, operating at the `l3_main_free_clk` interconnect clock frequency. For memory, read acceptance is two, write acceptance is two, and total acceptance is two with a round-robin arbitration.

Related Information

- **Error Checking and Correction Controller** on page 11-1
Error Checking and Correction (ECC) controllers provide single- and double-bit error memory protection for integrated on-chip RAM and peripheral RAMs within the hard processor system (HPS).
- **SoC Security** on page 6-1

Functional Description of the On-Chip RAM

The on-chip RAM uses a 64-bit slave interface. The slave interface supports transfers between memory and the L3 interconnect. All reads and writes are serviced in order.

The on-chip RAM has an exclusive monitor, as well. To ensure mutually exclusive access to shared data, use the exclusive access support built into the on-chip RAM. The on-chip RAM contains five monitors. Each monitor can be used by any of the following:

- CPU and CPU1
- FPGA2SOC
- FPGA2SDRAM0
- FPGA2SDRAM1
- FPGA2SDRAM2

Note: The on-chip RAM exclusive monitor for the MPU does not differentiate between individual CPUs.

Related Information

Clock Manager on page 2-1

On-Chip RAM Clocks

The on-chip RAM is driven by the `l3_main_clk` interconnect clock.

There are two clocks to the on-chip RAM that must be enabled. The first clock is for the memory bus and the second clock is for the ECC register interface. The ECC register interface clock is the `l4_mp_clk`.

On-Chip RAM Resets

The contents of the RAM remain unchanged on a cold or warm reset. Reset only clears the state associated with the slave interface.

However, when a reset is initiated because the Security Manager receives a tamper detection assertion, the memory is scrambled and cleared.

Related Information

- **Error Checking and Correction Controller** on page 11-1
Error Checking and Correction (ECC) controllers provide single- and double-bit error memory protection for integrated on-chip RAM and peripheral RAMs within the hard processor system (HPS).
- **SoC Security** on page 6-1

On-Chip RAM Initialization

You must initialize the on-chip RAM before you enable the ECC. Failure to do so triggers spurious interrupts.

The on-chip RAM can be initialized by setting the `INITA` bit in the ECC Control Register (CTRL). After the Initialization Status (`INITSTAT`) register indicates the initialization has completed, the ECC Controller can be enabled and ECC interrupts can be configured.

Related Information

Error Checking and Correction Controller on page 11-1
For more information about ECC, refer to the *Error Checking and Correction Controller* chapter of the Arria 10 Device Handbook.

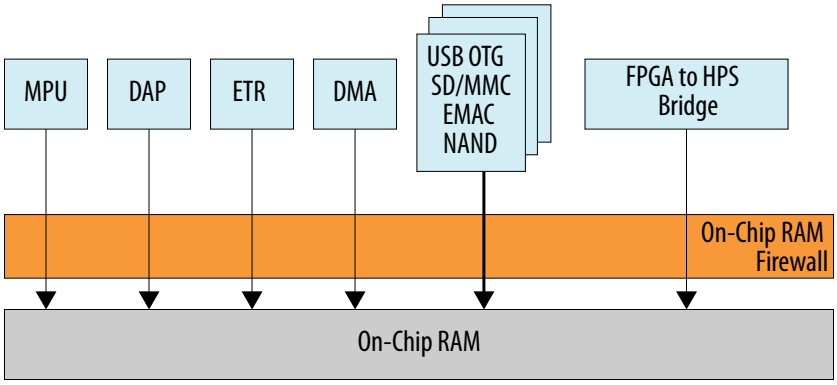
On-Chip RAM Firewall

At reset all memories are secure. Out of reset, regions of memories can be configured for non-secure accesses.

The on-chip RAM is divided into six regions of 4 KB. Within each region, a non-secure or shared memory region can be assigned by programming a `base` and `limit` value in the corresponding `regionNaddr` register, with `N` denoting the region number. Each of these regions can be enabled by writing to the `enable` register or setting the corresponding bit in the `enable_set` register.

When an incoming transaction falls within any enabled non-secure regions, the firewall allows both secure and non-secure packets. When the transaction is outside of any enabled regions, the firewall only allows secure packets.

Figure 12-2: On-Chip RAM Firewall



ECC Protection

The ECC controller operation and functionality is programmable through the ECC register slave interface, as shown in the **On-Chip RAM Block Diagram** in the "On-Chip RAM Block Diagram and System Integration" section. The ECC controller's register interface provides host access to configure the ECC logic as well as inject bit errors into the memory for testing purposes. It also provides host access to memory initialization hardware used to clear the memory contents, including the ECC bits. The ECC controller generates interrupts upon occurrences of single- and double-bit errors, and the interrupt signals are connected to the system manager.

Related Information

- **Error Checking and Correction Controller** on page 11-1
Error Checking and Correction (ECC) controllers provide single- and double-bit error memory protection for integrated on-chip RAM and peripheral RAMs within the hard processor system (HPS).
- **SoC Security** on page 6-1

Boot ROM

Features of the Boot ROM

The boot ROM offers the following features:

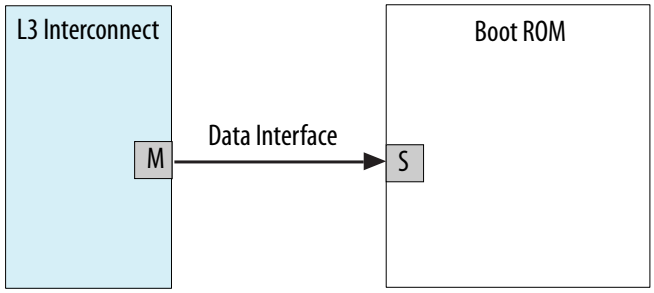
- 32-bit interface
- 128 KB size
- Capable of data transfers for every clock cycle
- Secure initialization

Related Information

Clock Manager on page 2-1

Boot ROM Block Diagram and System Integration

Figure 12-3: Block diagram of On-Chip ROM



Boot ROM and L3 interconnect transfers data through a 32-bit data interface that passes through a firewall, operating at the `l3_main_free_clk` interconnect clock frequency. The memory has a read acceptance of one, a write acceptance of one, and a total acceptance of two with round-robin arbitration.

Related Information

SoC Security on page 6-1

Functional Description of the Boot ROM

The boot ROM is used only for booting the system.

The boot ROM contains the execution code that identifies if there is a secure or non-secure boot, initializes the system to bring it out of reset, finds bootloader and can decrypt and/or authenticate it based on user fuses.

On a cold or warm reset of the microprocessor unit (MPU) subsystem, MPU0 executes the Preloader code stored in the boot ROM. The Preloader occurs after the initial boot stage, which is the boot ROM. It is loaded by the boot ROM and copied to the on-chip RAM. However, if you choose to boot the Preloader from the FPGA, or if a failure occurred at boot ROM, then as a fallback boot device, the Preloader can then run directly on the FPGA without the need to copy to on-chip RAM.

The boot ROM uses an 32-bit slave interface. The slave interface supports transfers between memory and the L3 interconnect. All writes return an error response.

Related Information

- **Clock Manager** on page 2-1
- **SoC Security** on page 6-1
- **Booting and Configuration** on page 30-1

Boot ROM Clocks

The boot ROM is driven by the `l3_main_clk` interconnect clock.

The boot ROM needs to set up the Main PLL and make sure it sources all of the system's clocks. The sequence applied for both Warm and Cold resets is similar. The PLL's are not set up on an FPGA boot, a successful RAM boot, or if the user selects a bypass mode.

Related Information

- **Clock Manager** on page 2-1
- **SoC Security** on page 6-1

Boot ROM Resets

Reset to on-chip ROM does not need to distinguish between warm and cold reset. Therefore, the content of the ROM will remain unchanged on a cold or warm reset.

On-Chip Memory Address Map and Register Definitions

For complete HPS address map and register definitions, refer to the **Arria 10 HPS Address Map and Register Definitions**.

Related Information

Error Checking and Correction Controller on page 11-1
For more information about how to program the ECC registers, refer to this chapter.

onchip_ram_ecc Address Map

Module Instance	Base Address	End Address
ecc_onchip_ram_ecc_register-Block	0xFF8C3000	0xFF8C7FFF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
IP_REV_ID on page 11-250	0x0	32	RO	0x0	
CTRL on page 11-251	0x8	32	RW	0x2	ECC Control Register
INITSTAT on page 11-252	0xC	32	RW	0x0	This bit is used to set the initialize the memory and ecc to a known value
ERRINTEN on page 11-252	0x10	32	RW	0x0	Error Interrupt enable
ERRINTENS on page 11-253	0x14	32	RW	0x0	Error Interrupt set

Register	Offset	Width	Access	Reset Value	Description
ERRINTENR on page 11-254	0x18	32	RW	0x0	Error Interrupt reset.
INTMODE on page 11-255	0x1C	32	RW	0x0	Reads reflect SERRINTEN.
INTSTAT on page 11-255	0x20	32	RW	0x0	This bit is used to enable interrupt generation on SERR lookup table overflow. When all the entries in the table are valid=1 and this is bit is enabled, serr_req signal will be asserted.
INTTEST on page 11-256	0x24	32	RW	0x0	This bits is used to test interrupt from ECC RAM to GIC
MODSTAT on page 11-257	0x28	32	RW	0x0	Counter feature status flag
DERRADDRA on page 11-258	0x2C	32	RO	0x0	This register shows the address of PORTA current double-bit error. RAM size will determine the maximum number of address bits.
SERRADDRA on page 11-258	0x30	32	RO	0x0	This register shows the address of PORTA current single-bit error. RAM size will determine the maximum number of address bits.
SERRCNTREG on page 11-259	0x3C	32	RW	0x0	Maximum counter value for single-bit error interrupt
ECC_Addrbus on page 11-260	0x40	32	RW	0x0	MSB bit of address is determined by ADR.
ECC_RData0bus on page 11-261	0x44	32	RO	0x0	Data will be read to this register field.
ECC_RData1bus on page 11-261	0x48	32	RO	0x0	Data will be read to this register field.

Register	Offset	Width	Access	Reset Value	Description
ECC_RData2bus on page 11-262	0x4C	32	RO	0x0	Data will be read to this register field.
ECC_RData3bus on page 11-263	0x50	32	RO	0x0	Data will be read to this register field.
ECC_WData0bus on page 11-263	0x54	32	WO	0x0	Data from the register will be written to the RAM.
ECC_WData1bus on page 11-264	0x58	32	WO	0x0	Data from the register will be written to the RAM.
ECC_WData2bus on page 11-265	0x5C	32	WO	0x0	Data from the register will be written to the RAM.
ECC_WData3bus on page 11-265	0x60	32	WO	0x0	Data from the register will be written to the RAM.
ECC_RDataecc0bus on page 11-266	0x64	32	RO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.
ECC_RDataecc1bus on page 11-267	0x68	32	RO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.
ECC_WDataecc0bus on page 11-268	0x6C	32	WO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Register	Offset	Width	Access	Reset Value	Description
ECC_WDataecc1bus on page 11-269	0x70	32	WO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.
ECC_dbytectrl on page 11-270	0x74	32	RW	0x0	Max number of implemented byte enabled is DAT/8
ECC_accctrl on page 11-271	0x78	32	RW	0x0	These bits determine which byte of data/ecc to write to RAM.
ECC_startacc on page 11-272	0x7C	32	RW	0x0	These bits determine which byte of data/ecc to write to RAM.
ECC_wdctrl on page 11-272	0x80	32	RW	0x0	Bits to Enable/Disable Watch Dog Timer
SERRLKUPA0 on page 11-273	0x90	32	RW	0x0	Single-bit error address in LOOKUP TABLE for PORTA.

onchip_ram_ecc Summary

Base Address: 0xFF8C3000

Register Address Offset	Bit Fields															
ecc_onchip_ram_ecc_register-Block																
IP_REV_ID 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SIREV															
	RO 0x0															

Register Address Offset	Bit Fields															
CTRL 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															INITA 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							CNT_RSTA 0x0	Reserved						ECC_SLVERR_DIS 0x1	ECC_EN 0x0
INITSTAT 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															INITCOM-PLETEA 0x0
ERRINTEN 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTEN 0x0
ERRINTENS 0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTS 0x0
ERRINTENR 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTR 0x0
INTMODE 0x1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															INTONCMP 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							INTONOVF 0x0	Reserved							INTMODE 0x0

Register Address Offset	Bit Fields															
INTSTAT 0x20	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							DERRPENA 0x0	Reserved							SERRPENA 0x0
INTTEST 0x24	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							TDERRA 0x0	Reserved							TSERRA 0x0
MODSTAT 0x28	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															CMPFLGA 0x0
DERRADDRA 0x2C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	Address 0x0														
SERRADDRA 0x30	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	Address 0x0														
SERRCNTREG 0x3C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	SERRCNT 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SERRCNT 0x0															

Register Address Offset	Bit Fields															
ECC_Addrbus 0x40	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_AddrBUS 0x0														
ECC_RData0bus 0x44	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_RData1bus 0x48	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_RData2bus 0x4C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_RData3bus 0x50	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_WData0bus 0x54	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															

Register Address Offset	Bit Fields															
ECC_WData1bus 0x58	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_WData2bus 0x5C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_WData3bus 0x60	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_RDataecc0bus 0x64	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved			ECC_RDataecc3BUS 0x0					Reserved			ECC_RDataecc2BUS 0x0				
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved			ECC_RDataecc1BUS 0x0					Reserved			ECC_RDataecc0BUS 0x0				
ECC_RDataecc1bus 0x68	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved			ECC_RDataecc7BUS 0x0					Reserved			ECC_RDataecc6BUS 0x0				
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved			ECC_RDataecc5BUS 0x0					Reserved			ECC_RDataecc4BUS 0x0				
ECC_WDataecc0bus 0x6C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved			ECC_WDataecc3BUS 0x0					Reserved			ECC_WDataecc2BUS 0x0				
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved			ECC_WDataecc1BUS 0x0					Reserved			ECC_WDataecc0BUS 0x0				

Register Address Offset	Bit Fields															
ECC_WDataecc1bus 0x70	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved			ECC_WDataecc7BUS 0x0					Reserved			ECC_WDataecc6BUS 0x0				
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved			ECC_WDataecc5BUS 0x0					Reserved			ECC_WDataecc4BUS 0x0				
ECC_dbytectrl 0x74	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								DBEN 0x0							
ECC_accctrl 0x78	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							RDWR 0x0	Reserved						ECCOVR 0x0	DATAOVR 0x0
ECC_startacc 0x7C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															ENBUSA 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															
ECC_wdctrl 0x80	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															WDEN_RAM 0x0
SERRLKUPA0 0x90	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	VALID RW 0x0	Reserved														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	Address RO 0x0														

IP_REV_ID

Module Instance	Base Address	Register Address
ecc_onchip_ram_ecc_register-Block	0xFF8C3000	0xFF8C3000

Offset: 0x0

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SIREV RO 0x0															

IP_REV_ID Fields

Bit	Name	Description	Access	Reset
15:0	SIREV	IP Rev # These bits indicate the silicon revision number.	RO	0x0

CTRL

ECC Control Register

Module Instance	Base Address	Register Address
ecc_onchip_ram_ecc_register-Block	0xFF8C3000	0xFF8C3008

Offset: 0x8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															INITA 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							CNT_RSTA 0x0	Reserved						ECC_SLVERR_DIS 0x1	ECC_EN 0x0

CTRL Fields

Bit	Name	Description	Access	Reset
16	INITA	Enable for the hardware memory initialization PORTA.	RW	0x0
8	CNT_RSTA	Enable to reset internal single-bit error counter A value to zero	RW	0x0
1	ECC_SLVERR_DIS	Enable to prevent double-bit ECC error from triggering SLVERR at ONCHIP_RAM AXI Interface RRESP output. Enable = 1'b1 Disable = 1'b0	RW	0x1
0	ECC_EN	Enable for the ECC detection and correction logic.	RW	0x0

INITSTAT

This bit is used to set the initialize the memory and ecc to a known value

Module Instance	Base Address	Register Address
ecc_onchip_ram_ecc_registerBlock	0xFF8C3000	0xFF8C300C

Offset: 0xC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															INITCOM-PLETEA 0x0

INITSTAT Fields

Bit	Name	Description	Access	Reset
0	INITCOMPLETEA	This bit is used to verify if the hardware memory initialization has completed PORTB.	RW	0x0

ERRINTEN

Error Interrupt enable

Module Instance	Base Address	Register Address
ecc_onchip_ram_ecc_registerBlock	0xFF8C3000	0xFF8C3010

Offset: 0x10

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															SERRINTEN 0x0

ERRINTEN Fields

Bit	Name	Description	Access	Reset
0	SERRINTEN	This bit is used to enable the single bit error interrupt of ECC RAM system	RW	0x0

ERRINTENS

Error Interrupt set

Module Instance	Base Address	Register Address
ecc_onchip_ram_ecc_register-Block	0xFF8C3000	0xFF8C3014

Offset: 0x14

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															SERRINTS 0x0

ERRINTENS Fields

Bit	Name	Description	Access	Reset
0	SERRINTS	This bit is used to set the single-bit error interrupt bit.	RW	0x0

ERRINTENR

Error Interrupt reset.

Module Instance	Base Address	Register Address
ecc_onchip_ram_ecc_register-Block	0xFF8C3000	0xFF8C3018

Offset: 0x18

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															SERRINTR 0x0

ERRINTENR Fields

Bit	Name	Description	Access	Reset
0	SERRINTR	This bit is used to reset the single-bit error interrupt bit. Reads reflect SERRINTEN. 1'b0: Writing of zero has no effect. 1'b1: By writing one, this bit will reset SERRINTEN bit to 0. This is performing a bitwise writing of this feature.	RW	0x0

INTMODE

Reads reflect SERRINTEN.

Module Instance	Base Address	Register Address
ecc_onchip_ram_ecc_register-Block	0xFF8C3000	0xFF8C301C

Offset: 0x1C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															INTONCMP 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							INTONOVF 0x0	Reserved							INTMODE 0x0

INTMODE Fields

Bit	Name	Description	Access	Reset
16	INTONCMP	Enable interrupt on compare.	RW	0x0
8	INTONOVF	Enable interrupt on overflow.	RW	0x0
0	INTMODE	Interrupt mode for single-bit errors.	RW	0x0

INTSTAT

This bit is used to enable interrupt generation on SERR lookup table overflow. When all the entries in the table are valid=1 and this is bit is enabled, serr_req signal will be asserted.

Module Instance	Base Address	Register Address
ecc_onchip_ram_ecc_register-Block	0xFF8C3000	0xFF8C3020

Offset: 0x20

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							DERRPENA 0x0	Reserved							SERRPENA 0x0

INTSTAT Fields

Bit	Name	Description	Access	Reset
8	DERRPENA	Double-bit error pending for PORTA.	RW	0x0
0	SERRPENA	Single-bit error pending for PORTA.	RW	0x0

INTTEST

This bits is used to test interrupt from ECC RAM to GIC

Module Instance	Base Address	Register Address
ecc_onchip_ram_ecc_register-Block	0xFF8C3000	0xFF8C3024

Offset: 0x24

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							TDERRA 0x0	Reserved							TSERRA 0x0

INTTEST Fields

Bit	Name	Description	Access	Reset
8	TDERRA	Test PORTA Double-bit error.	RW	0x0
0	TSERRA	Test PORTA Single-bit error.	RW	0x0

MODSTAT

Counter feature status flag

Module Instance	Base Address	Register Address
ecc_onchip_ram_ecc_register-Block	0xFF8C3000	0xFF8C3028

Offset: 0x28

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															CMPFLGA 0x0

MODSTAT Fields

Bit	Name	Description	Access	Reset
0	CMPFLGA	Port A compare status flag	RW	0x0

DERRADDRA

This register shows the address of PORTA current double-bit error. RAM size will determine the maximum number of address bits.

Module Instance	Base Address	Register Address
ecc_onchip_ram_ecc_register-Block	0xFF8C3000	0xFF8C302C

Offset: 0x2C

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	Address 0x0														

DERRADDRA Fields

Bit	Name	Description	Access	Reset
14:0	Address	Recent double-bit error address.	RO	0x0

SERRADDRA

This register shows the address of PORTA current single-bit error. RAM size will determine the maximum number of address bits.

Module Instance	Base Address	Register Address
ecc_onchip_ram_ecc_register-Block	0xFF8C3000	0xFF8C3030

Offset: 0x30

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	Address 0x0														

SERRADDRA Fields

Bit	Name	Description	Access	Reset
14:0	Address	Recent single-bit error address.	RO	0x0

SERRCNTREG

Maximum counter value for single-bit error interrupt

Module Instance	Base Address	Register Address
ecc_onchip_ram_ecc_register-Block	0xFF8C3000	0xFF8C303C

Offset: 0x3C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SERRCNT 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SERRCNT 0x0															

SERRCNTREG Fields

Bit	Name	Description	Access	Reset
31:0	SERRCNT	Counter value	RW	0x0

ECC_Addrbus

MSB bit of address is determined by ADR.

Module Instance	Base Address	Register Address
ecc_onchip_ram_ecc_register-Block	0xFF8C3000	0xFF8C3040

Offset: 0x40

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	ECC_AddrBUS 0x0														

ECC_Addrbus Fields

Bit	Name	Description	Access	Reset
14:0	ECC_AddrBUS	Address will be driven to RAM to either read or write the data. Address will be latched by the RAM when the Enbus is asserted.	RW	0x0

ECC_RData0bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_onchip_ram_ecc_register-Block	0xFF8C3000	0xFF8C3044

Offset: 0x44

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData0bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_RDataBUS	ECC_RDataBUS[31:0].	RO	0x0

ECC_RData1bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_onchip_ram_ecc_register-Block	0xFF8C3000	0xFF8C3048

Offset: 0x48

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData1bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_RDataBUS	ECC_RDataBUS[63:32].	RO	0x0

ECC_RData2bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_onchip_ram_ecc_register-Block	0xFF8C3000	0xFF8C304C

Offset: 0x4C

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData2bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_RDataBUS	ECC_RDataBUS[95:64].	RO	0x0

ECC_RData3bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_onchip_ram_ecc_register-Block	0xFF8C3000	0xFF8C3050

Offset: 0x50

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData3bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_RDataBUS	ECC_RDataBUS[127-96].	RO	0x0

ECC_WData0bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_onchip_ram_ecc_register-Block	0xFF8C3000	0xFF8C3054

Offset: 0x54

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData0bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_WDataBUS	ECC_WDataBUS[31:0].	WO	0x0

ECC_WData1bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_onchip_ram_ecc_register-Block	0xFF8C3000	0xFF8C3058

Offset: 0x58

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData1bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_WDataBUS	ECC_WDataBUS[63:32].	WO	0x0

ECC_WData2bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_onchip_ram_ecc_register-Block	0xFF8C3000	0xFF8C305C

Offset: 0x5C

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData2bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_WDataBUS	ECC_WDataBUS[95-64].	WO	0x0

ECC_WData3bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_onchip_ram_ecc_register-Block	0xFF8C3000	0xFF8C3060

Offset: 0x60

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData3bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_WDataBUS	ECC_WDataBUS[127-96].	WO	0x0

ECC_RDataecc0bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_onchip_ram_ecc_register-Block	0xFF8C3000	0xFF8C3064

Offset: 0x64

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved			ECC_RDataecc3BUS 0x0					Reserved			ECC_RDataecc2BUS 0x0				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			ECC_RDataecc1BUS 0x0					Reserved			ECC_RDataecc0BUS 0x0				

ECC_RDataecc0bus Fields

Bit	Name	Description	Access	Reset
28:24	ECC_RDataecc3BUS	Eccdata will be read to this register field.	RO	0x0
20:16	ECC_RDataecc2BUS	Eccdata will be read to this register field.	RO	0x0
12:8	ECC_RDataecc1BUS	Eccdata will be read to this register field.	RO	0x0
4:0	ECC_RDataecc0BUS	Eccdata will be read to this register field.	RO	0x0

ECC_RDataecc1bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_onchip_ram_ecc_register-Block	0xFF8C3000	0xFF8C3068

Offset: 0x68

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved			ECC_RDataecc7BUS 0x0					Reserved			ECC_RDataecc6BUS 0x0				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			ECC_RDataecc5BUS 0x0					Reserved			ECC_RDataecc4BUS 0x0				

ECC_RDataecc1bus Fields

Bit	Name	Description	Access	Reset
28:24	ECC_RDataecc7BUS	Eccdata will be read to this register field.	RO	0x0
20:16	ECC_RDataecc6BUS	Eccdata will be read to this register field.	RO	0x0
12:8	ECC_RDataecc5BUS	Eccdata will be read to this register field.	RO	0x0
4:0	ECC_RDataecc4BUS	Eccdata will be read to this register field.	RO	0x0

ECC_WDataecc0bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_onchip_ram_ecc_register-Block	0xFF8C3000	0xFF8C306C

Offset: 0x6C

Access: WO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved			ECC_WDataecc3BUS 0x0					Reserved			ECC_WDataecc2BUS 0x0				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			ECC_WDataecc1BUS 0x0					Reserved			ECC_WDataecc0BUS 0x0				

ECC_WDataecc0bus Fields

Bit	Name	Description	Access	Reset
28:24	ECC_WDataecc3BUS	Eccdata from the register will be written to the RAM.	WO	0x0
20:16	ECC_WDataecc2BUS	Eccdata from the register will be written to the RAM.	WO	0x0
12:8	ECC_WDataecc1BUS	Eccdata from the register will be written to the RAM.	WO	0x0
4:0	ECC_WDataecc0BUS	Eccdata from the register will be written to the RAM.	WO	0x0

ECC_WDataecc1bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_onchip_ram_ecc_register-Block	0xFF8C3000	0xFF8C3070

Offset: 0x70

Access: WO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved			ECC_WDataecc7BUS 0x0					Reserved			ECC_WDataecc6BUS 0x0				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			ECC_WDataecc5BUS 0x0					Reserved			ECC_WDataecc4BUS 0x0				

ECC_WDataecc1bus Fields

Bit	Name	Description	Access	Reset
28:24	ECC_WDataecc7BUS	Eccdata from the register will be written to the RAM.	WO	0x0
20:16	ECC_WDataecc6BUS	Eccdata from the register will be written to the RAM.	WO	0x0
12:8	ECC_WDataecc5BUS	Eccdata from the register will be written to the RAM.	WO	0x0
4:0	ECC_WDataecc4BUS	Eccdata from the register will be written to the RAM.	WO	0x0

ECC_dbytectrl

ECC Data Byte Control. The maximum number of implemented byte enables equals the data width of the RAM divided by 8.

Module Instance	Base Address	Register Address
ecc_onchip_ram_ecc_register-Block	0xFF8C3000	0xFF8C3074

Offset: 0x74

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								DBEN 0x0							

ECC_dbytectrl Fields

Bit	Name	Description	Access	Reset
7:0	DBEN	Byte or word enable for access.	RW	0x0

ECC_accctrl

These bits determine which byte of data/ecc to write to RAM.

Module Instance	Base Address	Register Address
ecc_onchip_ram_ecc_register-Block	0xFF8C3000	0xFF8C3078

Offset: 0x78

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							RDWR 0x0	Reserved						ECCOVR 0x0	DATAOVR 0x0

ECC_accctrl Fields

Bit	Name	Description	Access	Reset
8	RDWR	Control for read/write.	RW	0x0
1	ECCOVR	ECC Data Override.	RW	0x0
0	DATAOVR	RAM Data Override. Override the ECC_dataBUS register with RAM data in read mode set by ECC_RW. 1'b0: Data override disabled. 1'b1: Data override enabled.	RW	0x0

ECC_startacc

These bits determine which byte of data/ecc to write to RAM.

Module Instance	Base Address	Register Address
ecc_onchip_ram_ecc_register-Block	0xFF8C3000	0xFF8C307C

Offset: 0x7C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															ENBUSA 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															

ECC_startacc Fields

Bit	Name	Description	Access	Reset
16	ENBUSA	Start RAM access for PORTA.	RW	0x0

ECC_wdctrl

Bits to Enable/Disable Watch Dog Timer

Module Instance	Base Address	Register Address
ecc_onchip_ram_ecc_register-Block	0xFF8C3000	0xFF8C3080

Offset: 0x80

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															WDEN_RAM 0x0

ECC_wdctrl Fields

Bit	Name	Description	Access	Reset
0	WDEN_RAM	Enable watchdog timeout for OCP register access to IP RAM.	RW	0x0

SERRLKUPA0

Single-bit error address in LOOKUP TABLE for PORTA.

Module Instance	Base Address	Register Address
ecc_onchip_ram_ecc_register-Block	0xFF8C3000	0xFF8C3090

Offset: 0x90

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VALID RW 0x0	Reserved														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	Address RO 0x0														

SERRLKUPA0 Fields

Bit	Name	Description	Access	Reset
31	VALID	Valid flag bit. Valid bit indicates if the address in this register is current or stale.	RW	0x0
14:0	Address	Recent Single-bit error address. This register shows the address of the each single-bit error. RAM size will determine the maximum number of address bits. If ram size is 32 Kbytes, bit 30-16 will be reserved and read as zero.	RO	0x0

ram_onchip_ram_block Address Map

This is the address space allocated to the on-chip RAM. The on-chip RAM can be used by the HPS for storing data or user code.

Module Instance	Base Address	End Address
i_ram_onchip_ram_block	0xFFE00000	0xFFFBFFFF

rom_onchip_rom_block Address Map

This address range is allocated for the boot ROM.

Module Instance	Base Address	End Address
i_rom_onchip_rom_block	0xFFFC0000	0xFFFFBFFF

Document Revision History

Document Revision History

Date	Version	Changes
October 2016	2016.10.28	Maintenance release
May 2016	2016.05.27	Maintenance release
May 2016	2016.05.03	Maintenance release
November 2015	2015.11.02	Maintenance release
May 2015	2015.05.04	Maintenance release

Date	Version	Changes
December 2014	2014.12.15	Maintenance release
August 2014	2014.08.18	Initial release

NAND Flash Controller 13

2016.10.28

a10_5v4 Subscribe Send Feedback

The hard processor system (HPS) provides a NAND flash controller to interface with external NAND flash memory in Altera system-on-a-chip (SoC) systems. You can use external flash memory to store software, or as extra storage capacity for large applications or user data. The HPS NAND flash controller is based on the Cadence Design IP® NAND Flash Memory Controller.

NAND Flash Controller Features

The NAND flash controller provides the following functionality and features:

- Supports one x8 or x16 NAND flash device

  Note: The HPS supports booting only from a x8 NAND flash device.
- Supports Open NAND Flash Interface (ONFI) 1.0
- Supports NAND flash memories from Hynix, Samsung, Toshiba, Micron, and STMicroelectronics
- Supports error correction codes (ECCs) providing single-bit error correction and double-bit error detection, with:
  - Sector size programmable 512 byte (4-, 8-, or 16-bit correction) or 1024 byte (24-bit correction)
  - Three NAND FIFOs - ECC Buffer, write FIFO and read FIFO
- Supports pipeline read-ahead and write commands for enhanced read and write throughput
- Supports devices with 32, 64, 128, 256, 384, or 512 pages per block
- Supports multiplane devices
- Supports page sizes of 512 bytes, 2 kilobytes (KB), 4 KB, or 8 KB
- Supports single layer cell (SLC) and multiple layer cell (MLC) devices with programmable correction capabilities
- Provides internal direct memory access (DMA)
- Provides programmable access timing

Related Information

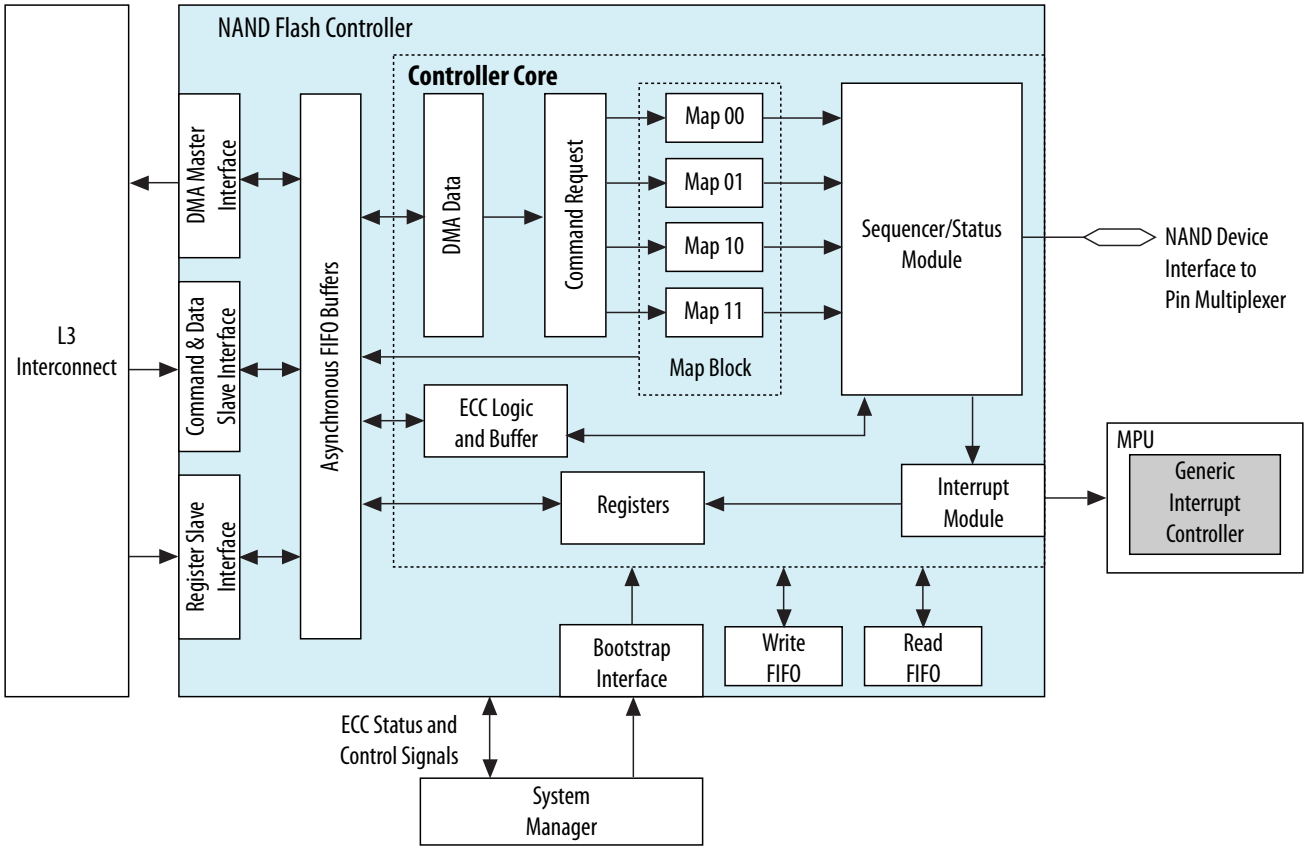
Supported Flash Devices for Arria 10 SoC

For more information, refer to the supported NAND flash devices section on this page.

NAND Flash Controller Block Diagram and System Integration

Figure 13-1: NAND Flash Controller Block Diagram

The following figure shows integration of the NAND flash controller in the HPS.



Features of the flash controller:

- Receives commands and data from the host through memory-mapped control and data registers connected to the command and data slave interface
- The host accesses the flash controller's control and status registers (CSRs) through the register slave interface.
- Handles all command sequencing and flash device interactions
- Generates interrupts to the HPS Cortex-A9 MPCore
- The DMA master interface provides accesses to and from the flash controller through the controller's built-in DMA.

NAND Flash Controller Signal Descriptions

All NAND pins have to be from one of the following categories:

- HPS Dedicated
- Shared
- FPGA

The following table lists all NAND Flash Interface signals available to both the HPS and FPGA.

Pins	Supported Data Width	Supported Number of CE and R/B
HPS Dedicated Pins	x8	1
Shared Pins	x8/x16	1
FPGA Pins	x8/x16	1 – 4

The type of pins you use determines the number of Chip Enable (CE) and Ready/Busy (RB) pairs available to you for use. For example, if you use HPS dedicated pins or shared pins, you can only use one CE and R/B pair. If you use FPGA pins, you can use multiple CE and R/B pairs.

Note: The options are mutually exclusive, which means you cannot use HPS dedicated pins, and route the CE and R/B signals to FPGA dedicated pins.

For more information on which signals route to the FPGA and HPS I/O, please refer to the *HPS Component Interface* chapter.

Table 13-1: NAND Flash Interface Signals

Qsys Port Name	Connected to FPGA	Connected to HPS I/O	HPS Pin Name
nand_adq_i[15:0]	Yes	Yes	NAND_ADQ[15:0]
nand_adq_oe	Yes	Yes	
nand_adq_o[15:0]	Yes	Yes	
nand_ale_o	Yes	Yes	NAND_ALE
nand_ce_o[3:0]	Yes, 4 chip enables	Yes, 1 chip enable	NAND_CE_N
nand_cle_o	Yes	Yes	NAND_CLE
nand_re_o	Yes	Yes	NAND_RE_N
nand_rdy_busy_i[3:0]	Yes, 4 ready/busy signals	Yes, 1 ready/busy signal	NAND_RB
nand_we_o	Yes	Yes	NAND_WE_N
nand_wp_o	Yes	Yes	NAND_WP_N

Related Information

HPS Component Interfaces on page 28-7
For more information about NAND Flash Controller signal routing to the FPGA and HPS I/O, refer to this chapter.

Functional Description of the NAND Flash Controller

This section describes the functionality of the NAND flash controller.

Discovery and Initialization

The NAND flash controller performs a specific initialization sequence after the HPS receives power and the flash device is stable. During initialization, the flash controller queries the flash device and configures itself according to one of the following flash device types:

- ONFI 1.0-compliant devices
- Legacy (non-ONFI) NAND devices

The NAND flash controller identifies ONFI-compliant connected devices using ONFI discovery protocol, by sending the `Read ID` command. For devices that do not recognize this command (especially for 512-byte page size devices), software must write to the system manager to assert the `bootstrap_512B_device` signal to identify the device type before releasing the NAND controller from reset.

To support booting and initialization, the `rdy_busy_in` pin must be connected.

The NAND flash controller performs the following initialization steps:

1. If the system manager is asserting `bootstrap_inhibit_init`, the flash controller goes directly to **step 7**.
2. When the device is ready, the flash controller sends the "Read ID" command to read the ONFI signature from the memory device, to determine whether an ONFI or a legacy device is connected.
3. If the data returned by the memory device has an ONFI signature, the flash controller then reads the device parameter page. The flash controller stores the relevant device feature information in internal memory control registers, enabling it to correctly program other registers in the flash device, and goes to **step 5**.
4. If the data does not have a valid ONFI signature, the flash controller assumes that it is a legacy (non-ONFI) device. The flash controller then performs the following steps:
   a. Sends the `reset` command to the device
   b. Reads the device signature information
   c. Stores the relevant values into internal memory controller registers
5. The flash controller resets the memory device. At the same time, it verifies the width of the memory interface. The HPS supports one 8-bit or 16-bit NAND flash device. The flash controller detects the memory interface width.
6. The flash controller sends the `Page Load` command to block 0, page 0 of the device, configuring direct read access, so the processor can boot from that page. The processor can start reading from the first page of the flash memory, which is the expected location of the pre-loader software.

   Note: The system manager can bypass this step by asserting `bootstrap_inhibit_b0p0_load` before reset is de-asserted.
7. The flash controller sends the `reset` command to the flash.
8. The flash controller clears the `rst_comp` bit in the `intr_status0` register in the `status` group to indicate to software that the flash reset is complete.

Bootstrap Interface

The NAND flash controller provides a bootstrap interface that allows software to override the default behavior of the flash controller. The bootstrap interface contains four bits, which when set appropriately, allows the flash controller to skip the initialization phase and begin loading from flash memory immediately after reset. These bits are driven by software through the system manager. They are sampled by the NAND flash controller when the controller is released from reset.

Related Information

System Manager on page 5-1
For more information about the bootstrap interface control bits.

Bootstrap Setting Bits

The following table lists the relevant bootstrap setting bits, found in the system manager's NAND flash controller register group. As an example, this table also lists recommended bootstrap settings for a 512-byte page device.

Table 13-2: Bootstrap Setting Bits

Bit	Example Value for 512-Byte Page
`noinit`	1[24]
`page512`	1
`noloadb0p0`	1
`tworowaddr`	• 1—flash device supports two-cycle addressing • 0—flash device support three-cycle addressing

Related Information

Configuration by Host on page 13-5

Configuration by Host

If the system manager sets `bootstrap_inhibit_init` to 1, the NAND flash controller does not perform the process described in "Discovery and Initialization". In this case, the host processor must configure the flash controller.

When performance is not a concern in the design, the timing registers can be left unprogrammed.

Related Information

- **Bootstrap Setting Bits** on page 13-5
  For recommended configuration-by-host settings to enable the basic read, write, and erase operations for a single-plane, 512 bytes/page device.
- **Discovery and Initialization** on page 13-4

[24] When this register is set, the NAND flash controller expects the host to program the related device parameter registers. For more information, refer to "Configuration by Host".

Recommended Bootstrap Settings for 512-Byte Page Device

Table 13-3: Recommended Bootstrap Settings for an 8-bit, 512-Byte Page Device

Register	Value
devices_connected	1
device_width	0 indicating an 8-bit NAND flash device
number_of_planes	1 indicating a single-plane device
device_main_area_size	The value of this register must reflect the flash device's page main area size.
device_spare_area_size	The value of this register must reflect the flash device's page spare area size.
pages_per_block	The value of this register must reflect number of pages per block in the flash device.

NAND Page Main and Spare Areas

Each NAND page has a main area and a spare area. The main area is intended for data storage. The spare area is intended for ECC and maintenance data, such as wear leveling information. Each block consists of a group of pages.

The sizes of the main and spare areas, and the number of blocks in a page, depend on the specific NAND device connected to the NAND flash controller. Therefore, the device-dependent registers, `device_main_area_size`, `device_spare_area_size`, and `pages_per_block`, must be programmed to match the characteristics of the device.

If your software does not perform the discovery and initialization sequence, the software must include an alternative method to determine the correct value of the device-dependent registers. The HPS boot ROM code enables discovery and initialization by default (that is, `bootstrap_inhibit_init = 0`).

Local Memory Buffer

The NAND flash controller has three local SRAM memory buffers.

- The write FIFO buffer is a 128 × 32-bit memory (512 total bytes)
- The read FIFO buffer is a 32 × 32-bit memory (128 total bytes)
- The ECC buffer is a 96 × 16-bit memory (1536 total bytes)

Each of these memories is protected by ECC, and by interrupts for single and double-bit errors. The ECC block is integrated around a memory wrapper. It provides outputs to notify the system manager when single-bit correctable errors are detected (and corrected) and when double-bit uncorrectable errors are detected. The ECC logic also allows injection of single- and double-bit errors for test purposes. It must be initialized to enable the ECC function.

Related Information

Error Checking and Correction Controller on page 11-1
For more information about ECC, refer to the *Error Checking and Correction Controller* chapter of the Arria 10 Device Handbook.

Clocks

The NAND clock runs synchronously from the NOC clocks, which is always active. To minimize the number of clocks, Clock Manager outputs software managed enables to the USB, SPI Masters, QSPI and NAND peripherals. The software enable for NAND is `nand_clk_en` and is set to ENABLE by default. Also, in Boot Mode, `nand_clk_en` is active to ensure that all clocks are active if RAM is cleared for security.

Table 13-4: Clock Inputs to NAND Flash Controller

Clock Signal	Description
`nand_x_clk`	Clock for master and slave interfaces and the ECC sector buffer.
`nand_clk`	Clock for the NAND flash controller.

The frequency of `nand_x_clk` is four times the frequency of `nand_clk`.

Clock Generation

The clock manager sends the top level clock from the HPS.

The clock manager sends the 200 MHz clock, `l4_mp_clk`, to the NAND Flash Controller. This clock becomes the NAND reference clock called `nand_mp_clk`. The `nand_mp_clk` is divided by four and is used for input and output. Since the NAND places a 200 MHz limit on the clock, each of these four generated clocks are 50 MHz and called `nand_clk`.

Clock Enable

The `nand_mp_clk` and `nand_clk` clocks have enables.

Clock Switching

When you use clock switching, you must follow the following requirements:

- Ensure that there is no activity.
- Software must disable this module during the frequency switch and re-enable it after the frequency has changed.
- When clock switching is complete, the software must reconfigure the NAND initialization registers according to the new frequency before triggering any new transactions onto the flash interface.

Resets

When a tamper is detected, the reset manager sends a special signal to the NAND data RAM. This scrambles and clears the RAM. In addition, the RAM can be cleared during a cold or warm reset, depending on how fuse bits are programmed in the device.

Before the NAND flash controller comes out of the reset state, the pin multiplexers for the flash external interface must be configured.

Related Information

Reset Manager on page 3-1

Taking the NAND Flash Controller Out of Reset

When a cold or warm reset is issued in the HPS, the reset manager resets this module and holds it in reset until software releases it.

After the Cortex-A9 MPCore CPU boots, it can deassert the reset signal by clearing the appropriate bits in the reset manager's corresponding reset register. For details about reset registers, refer to "Module Reset Signals".

Related Information

- **Module Reset Signals** on page 3-8
- **Modules Requiring Software Deassert** on page 3-13

Indexed Addressing

The NAND flash controller uses indexed addressing to reduce the address span consumed by the flash controller.

Indirect addressing is implemented by two registers, accessed through the `nanddata` region, as described in the "Register Map for Indexed Addressing" section.

Register Map for Indexed Addressing

Indexed addressing uses registers in the nanddata region of the HPS memory map. The nanddata region consists of a control register and a variable-size register that allows direct access to flash memory, as detailed in the following table.

Table 13-5: Register Map for Indexed Addressing

Register Name	Offset Address	Usage
`Control`	0x0	Identifies the page of flash memory to be read or written. Software writes the 32-bit control information consisting of **map** command type, block, and page address. The upper four bits must be set to 0. For specific usage of the `Control` register, refer to "Command Mapping".
`Data`	0x10	The `Data` register is a page-size window into the NAND flash. By reading from or writing to locations starting at this offset, the software reads directly from or writes directly to the page and block of NAND flash memory specified by the `Control` register. The Data register is always addressed on 32-bit word boundaries, although the physical flash device has an 8-bit-wide data path.

Related Information

- **Command Mapping** on page 13-9
- **HPS Peripheral Region Address Map** on page 1-20

Indexed Addressing Host Usage

The host uses indexed addressing as follows:

1. Program the 32-bit index-address field into the `Control` register in the `nanddata` region. This action provides the flash address parameters to the NAND flash controller.
2. Perform a 32-bit read or write in the `Data` register.
3. Perform additional 32-bit reads and writes if they are in the same page and block in flash memory.

It is unnecessary to write to the control register for every data transfer if a group of data transfers targets the same page and block address. For example, you can write the control register at the beginning of a page with the block and page address, and then read or write the entire page by directing consecutive transactions to the `Data` register.

Command Mapping

The NAND flash controller supports several flash controller-specific MAP commands, providing an abstraction level for programming a NAND flash device. By using the MAP commands, you can avoid directly programming device-specific commands. Using this abstraction layer provides enhanced performance. Commands take multiple cycles to send off-chip. The MAP commands let you initiate commands and let the flash controller sequence them off-chip to the NAND device.

The NAND flash controller supports the following flash controller-specific MAP commands:

- MAP00 commands—boot-read or buffer read/write during read-modify-write operations
- MAP01 commands—memory arrays read/write
- MAP10 commands—NAND flash controller commands
- MAP11 commands—low-level direct access

MAP00 Commands

MAP00 commands access a page buffer in the NAND flash device. Addressing always begins at 0x0 and extends to the page size specified by the `device_main_area_size` and `device_spare_area_size` registers in the `config` group. You can use this command to perform a boot read. Use MAP00 commands in read-modify-write (RMW) operations to read or write any word in the buffer. MAP00 commands allow a direct data path to the page buffer in the device.

The host can access the page buffer directly using the MAP00 commands only if there are no other MAP01 or MAP10 commands active on the NAND flash controller.

MAP00 Command Format

Table 13-6: MAP00 Command Format with Address Mapping

The following table shows the format of a MAP00 command. This command is written to the Command register in the `nanddata` region.

Address Bits	Name	Description
31:28	(reserved)	Set to 0
27:26	`CMD_MAP`	Set to 0
25:13	(reserved)	Set to 0

Address Bits	Name	Description
12:2	BUFF_ADDR	Data width-aligned buffer address on the memory device. Maximum page access is 8 KB.
1:0	(reserved)	Set to 0

MAP00 Usage Limitations

The usage of these commands under normal operations is limited to the following situations:

- They can be used to perform an Execute-in-Place (XIP) boot from the device; reading directly from the page buffer while booting directly from the device.
- MAP00 commands can be used to perform RMW operations where MAP00 writes are used to modify a read page in the device page buffer. Because the NAND flash controller does not perform ECC correction during such an operation, Altera does not recommend this method in an MLC device.
- In association with MAP11 commands, MAP00 commands provide a way for the host to directly access the device bypassing the hardware abstractions provided by NAND flash controller with MAP01 and MAP10 commands. This method is also used for debugging, or for issuing an operation that the flash controller might not support with MAP01 or MAP10 commands.

Restrictions:

- MAP00 commands cannot be used with MAP01 commands to read part of a page. Accesses using MAP01 commands must perform a complete page transfer.
- No ECC is performed during a MAP00 data access.
- DMA must be disabled (the `flag` bit of the `dma_enable` register in the `dma` group must be set to 0) while performing MAP00 operations.

MAP01 Commands

MAP01 commands transfer complete pages between the host memory and a specific page of the NAND flash device. Because the MAP01 commands support only page addresses, the entire page must be read or written at once. The actual number of commands required depends on the size of the data transfer. The Command register points to the first page and block in the transfer. You do not change the Command register when you initiate subsequent transactions in the transfer, but only when the entire page is transferred.

When the NAND flash controller receives a read command, it issues a load operation on the device, waits for the load to complete, and then returns read data. Read data must be read from the start of the page to the end of the page.

Write data must be written from the start of the page to the end of the page. When the NAND flash controller receives confirmation of the transfer, it issues commands to program the data into the device.

The flash controller ignores the byte enables for read and write commands and transfers the entire data width.

MAP01 Command Format

Table 13-7: MAP01 Command Format with Address Mapping

The following table shows the format of a MAP01 command. This command is written to the Command register in the `nanddata` region.

Address Bits	Name	Description
31:28	(reserved)	Set to 0
27:26	CMD_MAP	Set to 1
25:24	(reserved)	Set to 0
23:<*M*>	BLK_ADDR	Block address in the device
(<*M*>-1):0	PAGE_ADDR	Page address in the device

Note: <*M*> depends on the number of pages per block in the device. <*M*> = ceil(log2(<*device pages per block*>)). Therefore, use the following values:

- 32 pages per block: <*M*>=5
- 64 pages per block: <*M*>=6
- 128 pages per block: <*M*>=7
- 256 pages per block: <*M*>=8
- 384 pages per block: <*M*>=9
- 512 pages per block: <*M*>=9

MAP01 Usage Limitations

Use the MAP01 command as follows:

- A complete page must be read or written using a MAP01 command. During such transfers, every transaction from the host must have the same block and page address. The NAND flash controller internally keeps track of how much data it reads or writes.
- MAP00 commands cannot be used in between using MAP01 commands for reading or writing a page.
- DMA must be disabled (the `flag` bit of the `dma_enable` register in the `dma` group must be set to 0) while the host is performing MAP01 operations directly. If the host issues MAP01 commands to the NAND flash controller while DMA is enabled, the flash controller discards the request and generates an `unsup_cmd` interrupt.

MAP10 Commands

MAP10 commands provide an interface to the control plane of the NAND flash controller. MAP10 commands control special functions of the flash device, such as erase, lock, unlock, copy back, and page spare area access. Data passed in this command pathway targets the NAND flash controller rather than the flash device. Unlike other command types, the data (input or output) related to these transactions does not affect the contents of the flash device. Rather, this data specifies and performs the exact commands of the flash controller. Only the lower 16 bits of the `Data` register contain the relevant information.

MAP10 Command Format

Table 13-8: MAP10 Command Format with Address Mapping

The following table shows the format of a MAP10 command. This command is written to the Command register in the `nanddata` region.

Address Bits	Name	Description
31:28	(reserved)	Set to 0
27:26	CMD_MAP	Set to 2
25:24	(reserved)	Set to 0
23:<*M*>	BLK_ADDR	Block address in the device
(<*M*>-1):0	PAGE_ADDR	Page address in the device

Note: <*M*> depends on the number of pages per block in the device, as follows:

- 32 pages per block: <*M*>=5
- 64 pages per block: <*M*>=6
- 128 pages per block: <*M*>=7
- 256 pages per block: <*M*>=8
- 384 pages per block: <*M*>=9
- 512 pages per block: <*M*>=9

MAP10 Operations

Table 13-9: MAP10 Operations

Command	Function
0x01	Sets block address for erase and initiates operation
0x10	Sets unlock start address
0x11	Sets unlock end address and initiates unlock
0x21	Initiates a lock of all blocks
0x31	Initiates a lock-tight of all blocks
0x41	Sets up for spare area access
0x42	Sets up for default area access
0x43	Sets up for main+spare area access
0x60	Loads page to the buffer for a RMW operation
0x61	Sets the destination address for the page buffer in RMW operation

Command	Function
0x62	Writes the page buffer for a RMW operation
0x1000	Sets copy source address
0x11<*PP*>	Sets copy destination address and initiates a copy of <*PP*> pages
0x20<*PP*>	Sets up a pipeline read-ahead of <*PP*> pages
0x21<*PP*>	Sets up a pipeline write of <*PP*> pages

MAP10 Usage Limitations

Use the MAP10 commands as follows:

- MAP10 commands should be used to issue commands to the controller, such as erase, copy-back, lock, or unlock.
- MAP10 pipeline commands should also be used to read or write consecutive multiple pages from the flash device within a device block boundary. The host must first issue a MAP10 pipeline read or write command and then issue MAP01 commands to do the actual data transfers. The MAP10 pipeline read or write command instructs the NAND flash controller to use high-performance commands such as cache or multiplane because the flash controller has knowledge of multiple consecutive pages to be read. The pages must not cross a block boundary. If a block boundary is crossed, the flash controller generates an unsupported command (`unsup_cmd`) interrupt and drops the command.
- Up to four pipeline read or write commands, at the same time, can be issued to the NAND flash controller.
- While the NAND flash controller is performing MAP10 pipeline read or write commands, DMA must be disabled (the `flag` bit of the `dma_enable` register in the `dma` group must be set to 0). DMA must be disabled because the host is directly transferring data from and to the flash device through the flash controller.

MAP11 Commands

MAP11 commands provide direct access to the NAND flash controller's address and control cycles, allowing software to issue the commands directly to the flash device using the Command and Data registers. The MAP11 command is useful if the flash device supports a device-specific command not included with standard flash commands. It can also be useful for low-level debugging.

MAP11 commands provide a direct control path to the flash device. These commands execute command, address, and data read and write cycles directly on the NAND device interface. The host can issue only single-beat accesses to the `nanddata` region while using MAP11 commands. The following are the usage requirements:

- Command, address, and write data values are placed in the `Data` register.
- Command and address cycles to the device must be a write transaction on the host bus.
- For data cycles, the type of transaction on the host bus (read/write) determines the data cycle type on the device interface.
- On a read, the returned data also appears in the `Data` register.
- The Control register encodes the control operation type.

MAP11 Control Format

Table 13-10: MAP11 Control Format with Address Mapping

The following table shows the format of a MAP11 command. This command is written to the Command register in the `nanddata` region.

Address Bits	Name	Description
31:28	(reserved)	Set to 0
27:26	`CMD_MAP`	Set to 3
25:2	(reserved)	Set to 0
1:0	`TYPE`	Sets the control type as follows: • 0 = Command cycle • 1 = Address cycle • 2 = Data Read/Write Cycle

MAP11 Usage Limitations

Use the MAP11 commands as follows:

- Use MAP11 commands only in special cases, for debugging or sending device-specific commands that are not supported by the NAND flash controller.
- DMA must be disabled before you use MAP11 operations.
- The host can use only single beat access transfers when using MAP11 commands.

Note: MAP11 commands provide direct, unstructured access to the NAND flash device. Incorrect use can lead to unpredictable behavior.

Data DMA

The DMA transfers data with minimal host involvement. Software initiates data DMA with the MAP10 command.

The `flag` bit of the `dma_enable` register in the `dma` group enables data DMA functionality. Only enable or disable this functionality when there are no active transactions pending in the NAND flash controller. When the DMA is enabled, the flash controller initiates one DMA transfer per MAP10 command over the DMA master interface. When the DMA is disabled, all operations with the flash controller occur through the memory-mapped `nanddata` region.

The NAND flash controller supports up to four outstanding DMA commands, and ignores additional DMA commands. If software issues more than four outstanding DMA commands, the flash controller issues the `unsup_cmd` interrupt. On receipt of a DMA command, the flash controller performs command sequencing to transfer the number of pages requested in the DMA command. The DMA master reads or writes page data from the system memory in programmed burst-length chunks. After the DMA command completes, the flash controller issues an interrupt, and starts working on the next queued DMA command.

Pipelining allows the NAND flash controller to optimize its performance while executing back-to-back commands of the same type.

With certain restrictions, non-DMA MAP10 commands can be issued to the NAND flash controller while the flash controller is servicing DMA transactions. MAP00, MAP01, and MAP11 commands cannot be issued while DMA mode is enabled because the flash controller is operating in an extremely tightly-coupled, high-performance data transfer mode. On receipt of erroneous commands (MAP00, MAP01 or MAP11), the flash controller issues an `unsup_cmd` interrupt to inform the host about the violating command.

Consider the following points when using the DMA:

- A data DMA command is a type of MAP10 command. This command is interpreted by the data DMA engine and not by the flash controller core.
- No MAP01, MAP00, or MAP11 commands are allowed when DMA is enabled.
- Before the flash controller can accept data DMA commands, DMA must be enabled by setting the `flag` bit of the `dma_enable` register in the `dma` group.
- When DMA is enabled and the DMA engine initiates data transfers, ECC can be enabled for as-needed data correction concurrent with the data transfer.
- MAP10 commands are used along with data movements similar to MAP01 commands.
- With the exception of data DMA commands and MAP10 pipeline read and write commands, all other MAP10 commands such as erase, lock, unlock, and copy-back are forwarded to the flash controller.
- At any time, up to four outstanding data DMA commands can be handled by flash controller. During multi-page operations, the DMA transfer must not cross a flash block boundary. If it does, the flash controller generates an unsupported command (`unsup_cmd`) interrupt and drops the command.
- Data DMA commands are typically multi-page read and write commands with an associated pointer in host memory. The multi-page data is transferred to or from the host memory starting from the host memory pointer.
- Data DMA uses the `flash_burst_length` register in the `dma` group to determine the burst length value to drive on the interconnect. The data DMA hardware does not account for the interconnect's boundary crossing restrictions. The host must initialize the starting host address so that the DMA master burst transaction does not cross a 4 KB boundary.

There are two methods for initiating a DMA transaction: the multitransaction DMA command, and the burst DMA command.

Multi-Transaction DMA Command

The NAND flash controller processes multitransaction DMA commands only if it receives all four command-data pairs in order. The flash controller responds to out-of-order commands with an `unsup_cmd` interrupt. The flash controller also responds with an `unsup_cmd` interrupt if sequenced commands are interleaved with other flash controller MAP commands.

To initiate DMA with a multitransaction DMA command, you send four command-data pairs to the NAND flash controller through the Control and Data registers in the `nanddata` region, as shown in "Command-Data Pair Formats".

Related Information

Command-Data Pair Formats on page 13-16

Command-Data Pair Formats

Table 13-11: Command-Data Pair 1

	31:28	27:26	25:24	23:<*M*>	(<*M*> – 1):0
Command	0x0	0x2	0x0	Block address	Page address

Note: <*M*> = ceil(log2(<*device pages per block*>)). Therefore, use the following values:

- 32 pages per block: <*M*>=5
- 64 pages per block: <*M*>=6
- 128 pages per block: <*M*>=7
- 256 pages per block: <*M*>=8
- 384 pages per block: <*M*>=9
- 512 pages per block: <*M*>=9

	31:16	15:12	11:8	7:0
Data	0x0	0x2	0x0 = Read 0x1 = Write	<*PP*>= Number of pages

Table 13-12: Command-Data Pair 2

	31:28	27:26	25:24	23:8	7:0
Command	0x0	0x2	0x0	Memory address high	0x0

	31:16	15:12	11:8	7:0
Data	0x0	0x2	0x2	0x0

Table 13-13: Command-Data Pair 3

	31:28	27:26	25:24	23:8	7:0
Command	0x0	0x2	0x0	Memory address low[26]	0x0

	31:16	15:12	11:8	7:0
Data	0x0	0x2	0x3	0x0

Table 13-14: Command-Data Pair 4

	31:28	27:26	25:24	23:17	16	15:8	7:0
Command	0x0	0x2	0x0	0x0	INT[27]	Burst length	0x0

[25] <*M*> depends on the number of pages per block in the device. For more information about <*M*>, see the Note at the bottom of this table.

[26] The buffer address in host memory, which must be aligned to 32 bits.

	31:28	27:26	25:24	23:17	16	15:8	7:0

Note: INT controls the value of the `dma_cmd_comp` bit of the `intr_status0` register in the `status` group at the end of the DMA transfer. INT can take on one of the following values:

- 0—Do not interrupt host. The `dma_cmd_comp` bit is set to 0.
- 1—Interrupt host. The `dma_cmd_comp` bit is set to 1.

	31:16	15:12	11:8	7:0
Data	0x0	0x2	0x4	0x0

Related Information

- **Indexed Addressing** on page 13-8
- **Burst DMA Command** on page 13-17

Using Multi-Transaction DMA Commands

If you want the NAND flash controller DMA to perform cacheable accesses then you must configure the cache bits by writing the `l3master` register in the `nandgrp` group in the system manager. The NAND flash controller DMA must be idle before you use the system manager to change its cache capabilities.

You can issue non-DMA MAP10 commands while the NAND flash controller is in DMA mode. For example, you might trigger a host-initiated page move between DMA commands, to achieve wear leveling. However, do not interleave non-DMA MAP10 commands between the command-data pairs in a set of multitransaction DMA commands. You must issue all four command-data pairs shown in the above tables before sending a different command.

Note: Do not issue MAP00, MAP01 or MAP11 commands while DMA is enabled.

MAP10 commands in multitransaction format are written to the `Data` register at offset 0x10 in `nanddata`, the same as MAP10 commands in increment four (INCR4) format (described in "Burst DMA Command").

Related Information

- **Indexed Addressing** on page 13-8
- **Burst DMA Command** on page 13-17
- **System Manager** on page 5-1

Burst DMA Command

You can initiate a DMA transfer by sending a command to the NAND flash controller as a burst transaction of four 16-bit accesses. This form of DMA command might be useful for initiating DMA transfers from custom IP in the FPGA fabric. Most processor cores cannot use this form of DMA command, because they cannot control the width of the burst.

When DMA is enabled, the NAND flash controller recognizes the MAP10 pipeline DMA command as an INCR4 command, in the format shown in the following table. The address decoding for MAP10 pipeline DMA command remains the same, as shown in "MAP10 Command Format".

(27) INT specifies the host interrupt to be generated at the end of the complete DMA transfer. For more information about INT, see the Note at the bottom of this table.

MAP10 commands in INCR4 format are written to the `Data` register at offset 0x10 in `nanddata`, the same as MAP10 commands in multitransaction format (described in the "Multi-Transaction DMA Command").

Table 13-15: MAP10 Burst DMA (INCR4) Command Structure

The following table lists the MAP10 burst DMA command structure. The burst DMA command carries the same information as the multi-transaction DMA command-data pairs, but in a very different format.

Data Beat	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Beat 0	0x2				0x0: read. 0x1: write.				<*PP*>=number of pages							
Beat 1[28]	Memory address high															
Beat 2[28]	Memory address low															
Beat 3	0x0							INT[29]	Burst length							

Note: INT controls the value of the `dma_cmd_comp` bit of the `intr_status0` register in the `status` group at the end of the DMA transfer. INT can take on one of the following values:

0—Do not interrupt host. The `dma_cmd_comp` bit is set to 0.

1—Interrupt host. The `dma_cmd_comp p` bit is set to 1.

You can optionally send the 16-bit fields in the above table to the NAND flash controller as four separate bursts of length 1 in sequential order. Altera recommends this method.

If you want the NAND flash controller DMA to perform cacheable accesses, you must configure the cache bits by writing the `l3master` register in the `nandgrp` group in the system manager. The NAND flash controller DMA must be idle before you use the system manager to modify its cache capabilities.

Related Information

- **Multi-Transaction DMA Command** on page 13-15
- **MAP10 Command Format** on page 13-11
- **System Manager** on page 5-1

ECC

The NAND flash controller incorporates ECC logic to calculate and correct bit errors. The flash controller uses a Bose-Chaudhuri-Hocquenghem (BCH) algorithm for detection of multiple errors in a page.

The NAND flash controller supports 512- and 1024-byte ECC sectors. The flash controller inserts ECC check bits for every 512 or 1024 bytes of data, depending on the selected sector size. After 512 or 1024 bytes, the flash controller writes the ECC check bit information to the device page.

ECC information is striped in between 512 or 1024 bytes of data across the page. The NAND flash controller reads ECC information in the same pattern and performs a calculation to check for the presence of errors.

[28] The buffer address in host memory, which must be aligned to 32 bits.

[29] INT specifies the host interrupt to be generated at the end of the complete DMA transfer. For more information about INT, see the Note at the bottom of this table.

Related Information

Memory Data Initialization on page 11-5
For more information about clearing memory data before enabling ECC, refer to "Memory Data Initialization" in the Error Checking and Correction section.

Correction Capability, Sector Size, and Check Bit Size

Table 13-16: Correction Capability, Sector Size, and Check Bit Size

Correction	Sector Size in Bytes	Check Bit Size in Bytes
4	512	8
8	512	14
16	512	26
24	1024	46

ECC Programming Modes

The NAND flash controller provides the following ECC programming modes that software uses to format a page:

- Main Area Transfer Mode
- Spare Area Transfer Mode
- Main+Spare Area Transfer Mode

Related Information

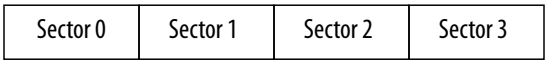
- **Main Area Transfer Mode** on page 13-19
- **Spare Area Transfer Mode** on page 13-19
- **Main+Spare Area Transfer Mode** on page 13-20

Main Area Transfer Mode

In main area transfer mode, when ECC is enabled, the NAND flash controller inserts ECC check bits in the data stream on writes and strips ECC check bits on reads. Software does not need to manage the ECC sectors when writing a page. ECC checking is performed by the flash controller, so software simply transfers the data.

If ECC is turned off, the NAND flash controller does not read or write ECC check bits.

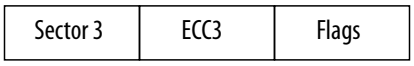
Figure 13-2: Main Area Transfer Mode for ECC



Spare Area Transfer Mode

The NAND flash controller does not introduce or interpret ECC check bits in spare area transfer mode, and acts as a pass-through for data transfer.

Figure 13-3: Spare Area Transfer Mode for ECC



Main+Spare Area Transfer Mode

In main+spare area transfer mode, the NAND flash controller expects software to format a page as shown in following figure. When ECC is enabled during a write operation, the flash controller-generated ECC check bits replace the ECC check bit data provided by software. During read operations, the flash controller forwards the ECC check bits from the device to the host. If ECC is disabled, page data received from the software is written to the device, and read data received from the device is forwarded to the host.

Figure 13-4: Main+Spare Area Transfer Mode for ECC

Preserving Bad Block Markers

When flash device manufacturers test their devices at the time of manufacture, they mark any bad device blocks that are found. Each bad block is marked at specific, known offsets, typically at the base of the spare area. A bad block marker is any byte value other than 0xFF (the normal state of erased flash).

Bad block markers can be overwritten by the last sector data in a page when ECC is enabled. This happens because the NAND flash controller also uses the main area of a page to store ECC information, which causes the last sector to spill over into the spare area. It is necessary for the system to preserve the bad block information prior to writing data, to ensure the correct identification of bad blocks in the flash device.

You can configure the NAND flash controller to skip over a specified number of bytes when it writes the last sector in a page to the spare area. This option allows the flash controller to preserve bad block markers. To use this option, write the desired offset to the `spare_area_skip_bytes` register in the `config` group. For example, if the device page size is 2 KB, and the device manufacturer stores the bad block markers in the first two bytes in the spare area, set the `spare_area_skip_bytes` register to 2. When the flash controller writes the last sector of the page that overlaps with the spare area, it starts at offset 2 in the spare area, skipping the bad block marker at offset 0. A value of 0 (default) specifies that no bytes are skipped. The value of `spare_area_skip_bytes` must be an even number. For example, if the bad block marker is a single byte, set `spare_area_skip_bytes` to 2.

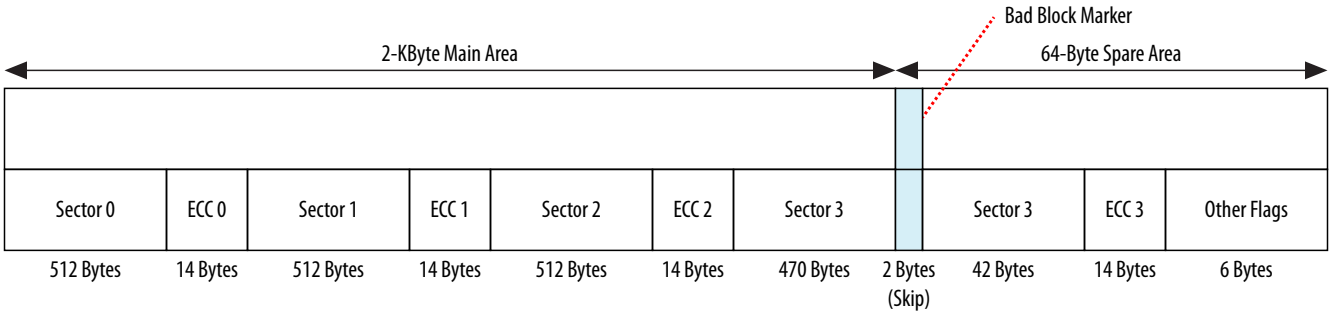
In main area transfer mode, the NAND flash controller does not skip the bad block marker. Instead, it overrides the bad block marker with the value programmed in the `spare_area_marker` register in the `config` group. This 8-bit register is used in conjunction with the `spare_area_skip_bytes` register in the `config` group to determine which bytes in the spare area of a page should be written with a the new marker value. For example, to mark a block as good set the `spare_area_marker` register to 0xFF and set the `spare_area_skip_bytes` register to the number of bytes that the marker should be written to, starting from the base of the spare area.

In the spare area transfer mode, the NAND flash controller ignores the `spare_area_skip_bytes` and `spare_area_marker` registers. The flash controller transfers the data exactly as received from the host or device.

In the main+spare area transfer mode, the NAND flash controller starts writing the last sector in a page into the spare area, starting at the offset specified in the `spare_area_skip_bytes` register. However, the area containing the bad block identifier information is overwritten by the data the host writes into the page. The host writes both the data sectors and the bad block markers. The flash controller depends on the host software to set up the bad block markers properly before writing the data.

Figure 13-5: Bad Block Marker

The following figure shows an example of how the NAND flash controller can skip over a bad block marker. In this example, the flash device has a 2-KB page with a 64-byte spare area. A 14-byte sector ECC is shown, with 8 byte per sector correction.



Related Information

Transfer Mode Operations on page 13-27
For detailed information about configuring the NAND flash controller for default, spare, or main+spare area transfer mode.

Error Correction Status

The ECC error correction information (`ECCCorInfo_b01`) register, in the `ecc` group, contains error correction information for each read or write that the NAND flash controller performs. The `ECCCorInfo_b01` register contains ECC error correction information in the `max_errors_b0` and `uncor_err_b0` fields.

At the end of data correction for the transaction in progress, `ECCCorInfo_b01` holds the maximum number of corrections applied to any ECC sector in the transaction. In addition, this register indicates whether the transaction as a whole has correctable errors, uncorrectable errors, or no errors at all. A transaction has no errors when none of the ECC sectors in the transaction has any errors. The transaction is marked as uncorrectable if any one of the sectors is uncorrectable. The transaction is marked as correctable if any one sector has correctable errors and none is uncorrectable.

At the end of each transaction, the host must read this register. The value of this register provides error data to the host about the block. The host can take corrective action after the number of correctable errors encountered reaches a particular threshold value.

NAND Flash Controller Programming Model

This section describes how the NAND flash controller is to be programmed by software running on the microprocessor unit (MPU).

Note: If you write a configuration register and follow it up with a data operation that is dependent on the value of this configuration register, Altera recommends that you read the value of the register before performing the data operation. This read operation ensures that the posted write of the register is completed and takes effect before the data operation is issued to the NAND flash controller.

Basic Flash Programming

This section describes the steps that must be taken by the software to access and control the NAND flash controller.

NAND Flash Controller Optimization Sequence

The software must configure the flash device for interrupt or polling mode, using the `bank0` bit of the `rb_pin_enabled` register in the `config` group. If the device is in polling mode, the software must also program the additional registers, to select the times and frequencies of the polling. Program the following registers in the `config` group:

- Set the `rb_pin_enabled` register to the desired mode of operation for each flash device.
- For polling mode, set the `load_wait_cnt` register to the appropriate value depending on the speed of operation of the NAND flash controller, and the desired wait value.
- For polling mode, set the `program_wait_cnt` register to the appropriate value by software depending on the speed of operation of the NAND flash controller, and the desired wait value.
- For polling mode, set the `erase_wait_cnt` register to the appropriate value by software depending on the speed of operation of the NAND flash controller, and the desired wait value.
- For polling mode, set the `int_mon_cyccnt` register to the appropriate value by software depending on the speed of operation of the NAND flash controller, and the desired wait value.

At any time, the software can change any flash device from interrupt mode to polling mode or vice-versa, using the `bank0` bit of the `rb_pin_enabled` register.

The software must ensure that the particular flash device does not have any outstanding transactions before changing the mode of operation for that particular flash device.

Device Initialization Sequence

At initialization, the host software must program the following registers in the `config` group:

- Set the `devices_connected` register to 1.
- Set the `device_width` register to 8.
- Set the `device_main_area_size` register to the appropriate value.
- Set the `device_spare_area_size` register to the appropriate value.
- Set the `pages_per_block` register according to the parameters of the flash device.
- Set the `number_of_planes` register according to the parameters of the flash device.
- If the device allows two ROW address cycles, the `flag` bit of the `two_row_addr_cycles` register must be set to 1. The host program can ensure this condition either of the following ways:
  - Set the `flag` bit of the `bootstrap_two_row_addr_cycles` register to 1 prior to the NAND flash controller's reset initialization sequence, causing the flash controller to initialize the bit automatically.
  - Set the `flag` bit of the `two_row_addr_cycles` register directly to 1.
- Clear the `chip_enable_dont_care` register in the `config` group to 0.

The NAND flash controller can identify the flash device features, allowing you to initialize the flash controller registers to interface correctly with the device, as described in *Discovery and Initialization*.

However, a few NAND devices do not follow any universally accepted identification protocol. If connected to such a device, the NAND flash controller cannot identify it correctly. If you are using such a device, your software must use other means to ensure that the initialization registers are set up correctly.

Related Information

Discovery and Initialization on page 13-4

Device Operation Control

This section provides a list of registers that you need to program while choosing to use multi-plane or cache operations on the device. If the device does not support multi-plane operations or cache operations, then these registers can be left at their power-on reset values with no impact on the functionality of the NAND flash controller. Even if the device supports these sequences, the software does not need to use them. Software can leave these registers at their power-on reset values.

Program the following registers in the `config` group to achieve the best performance from a given device:

- Set `flag` bit in the `multiplane_operation` register in the `config` group to 1 if the device supports multi-plane operations to access the data on the flash device connected to the NAND flash controller. If the flash controller is set up for multi-plane operations, the number of pages to be accessed is always a multiple of the number of planes in the device.
- If the NAND flash controller is configured for multi-plane operation, and if the device has support for multi-plane read command sequence, set the `multiplane_read_enable` register in the `config` group.
- If the device implements multiplane address restrictions, set the `flag` bit in the `multiplane_addr_restrict` register to 1.
- Initialize the `die_mask` and `first_block_of_next_plane` registers as per device requirements.
- If the device supports cache command sequences, enable the `cache_write_enable` and `cache_read_enable` registers in the `config` group.
- Clear the `flag` bit of the `copyback_disable` register in the `config` group to 0 if the device does not support the copyback command sequences. The register defaults to enabled state.
- The `read_mode`, `write_mode` and `copyback_mode` registers, in the `config` group, currently need not be written by software, because the NAND flash controller is capable of using the correct sequences based on a combination of some multi-plane or cache-related settings of the NAND flash controller and the manufacturer ID. If at some future time these settings change, program the registers to accommodate the change.

ECC Enabling

Before you start any data operation on the flash device, you must decide whether you want the ECC enabled or disabled.

To prevent spurious ECC errors, software must use the memory initialization block in the ECC controller to clear the entire memory data and initialize the ECC bits. The initialization block clears the memory data. Initializing the memory with the initialization block is independent of enabling the ECC.

If the ECC needs enabling, set up the appropriate correction level depending on the page size and the spare area available on the device.

Set the flag bit in the ecc_enable register in the config group to 1 to enable ECC. If enabled, the following registers in the config group must be programmed accordingly, else they can be ignored:

- Initialize the ecc_correction register to the appropriate correction level.
- Program the spare_area_skip_bytes and spare_area_marker registers in the config group if the software needs to preserve the bad block marker.

Related Information

ECC on page 13-18

NAND Flash Controller Performance Registers

These registers specify the size of the bursts on the device interface, which maximizes the overall performance on the NAND flash controller.

Initialize the flash_burst_length register in the dma group to a value which maximizes the performance of the device interface by minimizing the number of bursts required to transfer a page.

Interrupt and DMA Enabling

Prior to initiating any data operation on the NAND flash controller, the software must set appropriate interrupt status register bits. If the software uses the DMA logic in the flash controller, then the appropriate DMA enable and interrupts bits in the register space must be set.

1. Set the flag bit in the global_int_enable register in the config group to 1, to enable global interrupt.
2. Set the relevant bits of the intr_en0 register in the status group to 1 before initiating any operations if the flash controller is in interrupt mode. Altera recommends that the software reads back this register to ensure clearing an interrupt status. This recommendation applies also to an interrupt service routine.
3. Enable DMA if your application needs DMA mode. Enable DMA by setting the flag bit of the dma_enable register in the dma group. Altera recommends that the software reads back this register to ensure that the mode change is accepted before sending a DMA command to the flash controller.
4. If the DMA is enabled, then set up the appropriate bits of the dma_intr_en register in the dma group.

Order of Interrupt Status Bits Assertion

The following interrupt status bits, in the intr_status0 register in the status group, are listed in the order of interrupt bit setting:

1. time_out—All other interrupt bits are set to 0 when the watchdog time_out bit is asserted.
2. dma_cmd_comp—This bit signifies the completion of data transfer sequence.(30)
3. pipe_cpybck_cmd_comp—This bit is asserted when a copyback command or the last page of a pipeline command completes.
4. locked_blk—This bit is asserted when a program (or erase) is performed on a locked block.
5. INT_act—No relationship with other interrupt status bits. Indicates a transition from 0 to 1 on the ready_busy pin value for that flash device.
6. rst_comp—No relationship with other interrupt status bits. Occurs after a reset command has completed.
7. For an erase command:

(30) This interrupt status bit is the last to be asserted during a DMA operation to transfer data.

a. `erase_fail` (if failure)
b. `erase_comp`

8. For a program command:
   a. `locked_blk` (if performed on a locked block)
   b. `pipe_cmd_err` (if the pipeline sequence is broken by a MAP01 command)
   c. `page_xfer_inc` (at the end of each page data transfer)
   d. `program_fail` (if failure)
   e. `pipe_cpybck_cmd_comp`
   f. `program_comp`
   g. `dma_cmd_comp` (If DMA enabled)
9. For a read command:
   a. `pipe_cmd_err` (if the pipeline sequence is broken by a MAP01 command)
   b. `page_xfer_inc` (at the end of each page data transfer)
   c. `pipe_cpybck_cmd_comp`
   d. `load_comp`
   e. `ecc_uncor_error` (if failure)
   f. `dma_cmd_comp` (If DMA enabled)

Timing Registers

You must optimize the following registers for your flash device's speed grade and clock frequency. The NAND flash controller operates correctly with the power-on reset values. However, functioning with power-on reset values is a non-optimal mode that provides loose timing (large margins to the signals).

Set the following registers in the `config` group to optimize the NAND flash controller for the speed grade of the connected device and frequency of operation of the flash controller:

- `twhr2_and_we_2_re`
- `tcwaw_and_addr_2_data`
- `re_2_we`
- `acc_clks`
- `rdwr_en_lo_cnt`
- `rdwr_en_hi_cnt`
- `max_rd_delay`
- `cs_setup_cnt`
- `re_2_re`

Registers to Ignore

You do not need to initialize the following registers in the `config` group:

- The `transfer_spare_reg` register—Data transfer mode can be initialized using MAP10 commands.
- The `write_protect` register—Does not need initializing unless you are testing the write protection feature.

Flash-Related Special Function Operations

This section describes all the special functions that can be performed on the flash memory.

The functions are defined by MAP10 commands as described in *Command Mapping*.

Related Information

Command Mapping on page 13-9

Erase Operations

Before data can be written to flash, an erase cycle must occur. The NAND flash memory controller supports single block and multi-plane erases.

The controller decodes the block address from the indirect addressing shown in "MAP10 Command Format".

Related Information

MAP10 Command Format on page 13-11

Single Block Erase

A single command is needed to complete a single-block erase, as follows:

1. Write to the command register, setting the `CMD_MAP` field to 2 and the `BLK_ADDR` field to the desired erase block.
2. Write 0x01 to the `Data` register.

For a single block erase, the register `multiplane_operation` in the `config` group must be reset.

After the device completes the erase operation, the controller generates an `erase_comp` interrupt. If the erase operation fails, the `erase_fail` interrupt is issued. The failing block's address is updated in the `err_block_addr0` register in the `status` group.

Multi-Plane Erase

For multi-plane erases, the `number_of_planes` register in the `config` group holds the number of planes in the flash device, and the block address specified must be aligned to the number of planes in the device. The NAND flash controller consecutively erases each block of the memory, up to the number of planes available. Issue this command as follows:

1. Write to the command register, setting the `CMD_MAP` field to 2 and the `BLK_ADDR` field to the desired erase block.
2. Write 0x01 to the `Data` register.

For multi-plane erase, the register `multiplane_operation` in the `config` group must be set.

After the device completes erase operation on all planes, the NAND flash controller generates an `erase_comp` interrupt. If the erase operation fails on any of the blocks in a multi-plane erase command, an `erase_fail` interrupt is issued. The failing block's address is updated in the `err_block_addr0` register in the `status` group.

Lock Operations

The NAND flash controller supports the following features:

- Flash locking—The NAND flash controller supports all flash locking operations.

  The flash device itself might have limited support for these functions. If the device does not support locking functions, the flash controller ignores these commands.
- Lock-tight—With the lock-tight feature, the NAND flash controller can prevent lock status from being changed. After the memory is locked tight, the flash controller must be reset before any flash area can be locked or unlocked.

Unlocking a Span of Memory Blocks

To unlock several blocks of memory, perform the following steps:

1. Write to the command register, setting the `CMD_MAP` field to 2 and the `BLK_ADDR` field to the starting address of the area to unlock.
2. Write 0x10 to the `Data` register.
3. Write to the command register, setting the `CMD_MAP` field to 2 and the `BLK_ADDR` field to the ending address of the area to unlock.
4. Write 0x11 to the `Data` register.

When unlocking a range of blocks, the start block address must be less than the end block address. Otherwise, the NAND flash controller exhibits undetermined behavior.

Locking All Memory Blocks

To lock the entire memory:

1. Write to the command register, setting the `CMD_MAP` field to 2 and the `BLK_ADDR` field to any memory address.
2. Write 0x21 to the `Data` register.

Setting Lock-Tight on All Memory Blocks

After the lock-tight is applied, unlocked areas cannot be locked, and locked areas cannot be unlocked. To lock-tight the entire memory:

1. Write to the command register, setting the `CMD_MAP` field to 2 and the `BLK_ADDR` field to any memory address.
2. Write 0x31 to the `Data` register.

To disable the lock-tight, reset the memory controller.

Transfer Mode Operations

You can configure the NAND flash controller in one of the following modes of data transfer:

- Default area transfer mode
- Spare area transfer mode
- Main+spare area transfer mode

The NAND flash controller determines the default transfer mode from the setting of `transfer_spare_reg` register in the `config` group. Use MAP10 commands to dynamically change the transfer mode from the existing mode to the new mode. All subsequent commands are in the new mode of transfer. You must consider that transfer modes can be changed at logical data transfer boundaries. For example:

- At the beginning or end of a page in case of single page read or write.
- At the beginning or end of a complete multi-page pipeline read or write command.

transfer_spare_reg and MAP10 Transfer Mode Commands

The following table lists the functionality of the MAP10 transfer mode commands, and their mappings to the `transfer_spare_reg` register in the `config` group.

Table 13-17: transfer_spare_reg and MAP10 Transfer Mode Commands

transfer_spare_reg	MAP10 Transfer Mode Commands	Resulting NAND Flash Controller Mode
0	0x42	Main(31)
0	0x41	Spare
0	0x43	Main+spare
1	0x42	Main+spare(31)
1	0x41	Spare
1	0x43	Main+spare

Related Information

MAP10 Commands on page 13-11

Configure for Default Area Access

You only need to configure for default area access if the transfer mode was previously changed to spare area or main+spare area. To configure default area access:

1. Write to the command register, setting the `CMD_MAP` field to 2 and the `BLK_ADDR` field to any block.
2. Write 0x42 to the `Data` register.

The NAND flash controller determines the default area transfer mode from the setting of the `transfer_spare_reg` register in the `config` group. If it is set to 1, then the transfer mode becomes main +spare area, otherwise it is main area.

Configure for Spare Area Access

To access only the spare area of the flash device, use the MAP10 command to set up the NAND flash controller to read or write only the spare area on the device. After the flash controller is set up, use MAP01 read and write commands to access the spare area of the appropriate block and page addresses. To configure the NAND flash controller to access the spare area only, perform the following steps:

1. Write to the command register, setting the `CMD_MAP` field to 2 and the `BLK_ADDR` field to the target block.
2. Write 0x41 to the `Data` register.

Configure for Main+Spare Area Access

To configure the NAND flash controller to access the main+spare area:

1. Write to the command register, setting the `CMD_MAP` field to 2 and the `BLK_ ADDR` field to the target block.
2. Write 0x43 to the `Data` register.

(31) Default access mode (0x42) maps to either main (only) or main+spare mode, depending on the value of `transfer_spare_reg`.

Read-Modify-Write Operations

To read a specific page or modify a few words, bytes, or bits in a page, use the RMW operations. A read command copies the desired data from flash memory to a page buffer. You can then modify the information in the buffer using MAP00 buffer read and write commands and issue another command to write that information back to the memory.

The read-modify-write command operates on an entire page. This command is also useful for a copy type operation, where most of a page is saved to a new location. In this type of operation, the NAND flash controller reads the data, modifies a specified number of words in the page, and then writes the modified page to a new location.

Note: Because the data is modified within the page buffer of the flash device, the NAND flash controller ECC hardware is not used in RMW operations. Software must update the ECC during RMW operations.

Note: For a read-modify-write command to work with hardware ECC, the entire page must be read into system memory, modified, then written back to flash without relying on the RMW feature.

Read-Modify-Write Operation Flow

1. Start the flow by reading a page from the memory:
   - Write to the command register, setting the `CMD_MAP` field to 2 and the `BLK_ADDR` field to the starting address of the desired block.
   - Write 0x60 to the `Data` register.

   This step makes the page available to you in the page buffer in the flash device.
2. Provide the destination page address:
   - Write to the command register, setting the `CMD_MAP` field to 2 and the `BLK_ADDR` field to the destination address of the desired block.
   - Write 0x61 to the `Data` register.

   This step initiates the page program and provides the destination address to the device.
3. Use the MAP00 page buffer read and write commands to modify the data in the page buffer.
4. Write the page buffer data back to memory:
   - Write to the command register, setting the `CMD_MAP` field to 2 and the `BLK_ADDR` field to the same destination address.
   - Write 0x62 to the `Data` register.

   This step performs the write.

After the device completes the load operation, the NAND flash controller issues a `load_comp` interrupt. A `program_comp` interrupt is issued when the host issues the write command and the device completes the program operation.

If the page program operation (as a part of an RMW operation) results in a program failure in the device, `program_fail` interrupt is issued. The failing page's block and page address is updated in the `err_block_addr0` and `err_page_addr0` registers in the `status` group.

Copy-Back Operations

The NAND flash controller supports copy back operations. However, the flash device might have limited support for this function. If you attempt to perform a copy-back operation on a device that does not support copy-back, the NAND flash controller triggers an interrupt. An interrupt is also triggered if the

source block is not specified before the destination block is specified, or if the destination block is not specified in the next command following a source block specification.

The NAND flash controller cannot do ECC validation in case of copy-back commands. The flash controller copies the ECC data, but does not check it during the copy operation.

Note: Altera recommends that you use copy-back only if the ECC implemented in the flash controller is strong enough so that the next access can correct accumulated errors.

The 8-bit value <*PP*> specifies the number of pages for copy-back. With this feature, the NAND flash controller can copy multiple consecutive pages with a single command. When you issue a copy-back command, the flash controller performs the operation in the background. The flash controller puts other commands on hold until the current copy-back completes.

For a multi-plane device, if the `flag` bit in the `multiplane_operation` register in the `config` group is set to 1, multi-plane copy-back is available as an option. In this case, the block address specified must be plane-aligned and the value <*PP*> must specify the total number of pages to copy as a multiple of the number of planes. The block address continues incrementing, keeping the page address fixed, for the total number of planes in the device before incrementing the page address.

A `pipe_cpyback_cmd_comp` interrupt is generated when the flash controller has completed copy-back operation of all <*PP*> pages. If any page program operation (as a part of copy back operation) results in a program failure in the device, the `program_fail` interrupt is issued. The failing page's block and page address is updated in the `err_block_addr0` and `err_page_addr0` registers in the `status` group.

Copying a Memory Area (Single Plane)

To copy <*PP*> pages from one memory location to another:

1. Write to the command register, setting the `CMD_MAP` field to 2 and the `BLK_ADDR` field to the starting address of the area to be copied.
2. Write 0x1000 to the `Data` register.
3. Write to the command register, setting the `CMD_MAP` field to 2 and the `BLK_ADDR` field to the starting address of the new area to be written.
4. Write 0x11<*PP*> to the `Data` register, where <*PP*> is the number of pages to copy.

Copying a Memory Area (Multi-Plane)

To copy <*PP*> pages from one memory location to another:

1. Set the `flag` bit of the `multiplane_operation` register in the `config` group to 1.
2. Write to the command register, setting the `CMD_MAP` field to 2 and the `BLK_ADDR` field to the starting address of the area to be copied. The address must be plane-aligned.
3. Write 0x1000 to the `Data` register.
4. Write to the command register, setting the `CMD_MAP` field to 2 and the `BLK_ADDR` field to the starting address of the new area to be written. This address must also be plane-aligned.
5. Write 0x11<*PP*> to the `Data` register, where <*PP*> is the number of pages to copy.

The parameter <*PP*> must be a multiple of the number of planes in the device.

Pipeline Read-Ahead and Write-Ahead Operations

The NAND flash controller supports pipeline read-ahead and write-ahead operations. However, the flash device might have limited support for this function. If the device does not support pipeline read-ahead or write-ahead, the flash controller processes these commands as standard reads or writes.

The NAND flash controller can handle at the most four outstanding pipeline commands, queued up in the order in which the flash controller received the commands. The flash controller operates on the pipeline command at the head of the queue until all the pages corresponding to the pipeline command are executed. The flash controller then pops the pipeline command at the head of the queue and proceeds to work on the next pipeline command in the queue.

Pipeline Read-Ahead Function

The pipeline read-ahead function allows for a continuous reading of the flash memory. On receiving a pipeline read command, the flash controller immediately issues a load command to the device. While data is read out with MAP01 commands in a consecutive or multi-plane address pattern, the flash controller maintains additional cache or multi-plane read command sequencing for continuous streaming of data from the flash device.

Pipeline read-ahead commands can read data from the queue in this interleaved fashion. The parameter <PP> denotes the total number of pages in multiples of the number of planes available, and the block address must be plane-aligned, which keeps the page address constant while incrementing the block address for each page-size chunk of data. After reading from every plane, the NAND flash controller increments the page address and resets the block address to the initial address. You can also use pipeline write-ahead commands in multi-plane mode. The write operation works similarly to the read operation, holding the page address constant while incrementing the block address until all planes are written.

Note: The same four-entry queue is used to queue the address and page count for pipeline read-ahead and write-ahead commands. This commonality requires that you use MAP01 commands to read out all pages for a pipeline read-ahead command before the next pipeline command can be processed. Similarly, you must write to all pages pertaining to pipeline write-ahead command before the next pipeline command can be processed.

Because the value of the `flag` bit of the `multiplane_operation` register in the `config` group determines pipeline read-ahead or write-ahead behavior, it can only be changed when the pipeline registers are empty.

When the host issues a pipeline read-ahead command, and the flash controller is idle, the load operation occurs immediately.

Note: The read-ahead command does not return the data to the host, and the write-ahead command does not write data to the flash address. The NAND flash controller loads the read data. The read data is returned to the host only when the host issues MAP01 commands to read the data. Similarly, the flash controller loads the write data, and writes it to the flash only when the host issues MAP01 commands to write the data.

Set Up a Single Area for Pipeline Read-Ahead

To set up an area for pipeline read-ahead, perform the following steps:

1. Write to the command register, setting the `CMD_MAP` field to 2 and the `BLK_ADDR` field to the starting address of the block to pre-read.
2. Write 0x20<*PP*> to the `Data` register, where the 0 sets this command as a read-ahead and <*PP*> is the number of pages to pre-read. The pages must not cross a block boundary. If a block boundary is crossed, the NAND flash controller generates an unsupported command (`unsup_cmd`) interrupt and drops the command.

The read-ahead command is a hint to the flash device to start loading the next page in the page buffer as soon as the previous page buffer operation has completed. After you set up the read-ahead, use a MAP01

command to actually read the data. In the MAP01 command, specify the same starting address as in the read-ahead.

If the read command received following a pipeline read-ahead request is not to a pre-read page, then an interrupt bit is set to 1 and the pipeline read-ahead or write-ahead registers are cleared. You must issue a new pipeline read-ahead request to re-load the same data. You must use MAP01 commands to read all of the data that is pre-read before the NAND flash controller returns to the idle state.

Pipeline Write-Ahead Function

The pipeline write-ahead function allows for a continuous writing of the flash memory. While data is written with MAP01 commands in a consecutive or multi-plane address pattern, the NAND flash controller maintains cache or multi-plane command sequences for continuous streaming of data into the flash device.

For pipeline write commands, if any page program results in a failure in the device, a `program_fail` interrupt is issued. The failing page's block and page addresses are updated in the `err_block_addr0` and `err_page_addr0` registers in the `status` group.

Set Up a Single Area for Pipeline Write-Ahead

To set up an area for pipeline write-ahead:

1. Write to the command register, setting the `CMD_MAP` field to 2 and the `BLK_ADDR` field to the starting address of the block to pre-write.
2. Write 0x21<PP> to the `Data` register, where the value 1 sets this command as a write-ahead and <PP> is the number of pages to pre-write. The pages must not cross a block boundary. If a block boundary is crossed, the NAND flash controller generates an unsupported command (`unsup_cmd`) interrupt and drops the command.

After you set up the write-ahead, use a MAP01 command to write the data. In the MAP01 command, specify the same starting address as in the write-ahead.

If the write command received following a pipeline write-ahead request is not to a pre-written page, then an interrupt bit is set to 1 and the pipeline read-ahead or write-ahead registers are cleared. You must issue a new pipeline write-ahead request to configure the write logic.

You must use MAP01 commands to write all of the data that is pre-written before the NAND flash controller returns to the idle state.

Other Supported Commands

MAP01 commands must read or write pages in the same sequence that the pipelined commands were issued to the NAND flash controller. If the host issues multiple pipeline commands, pages must be read or written in the order the pipeline commands were issued. It is not possible to read or write pages for a second pipeline command before completing the first pipeline command. If the pipeline sequence is broken by a MAP01 command, the `pipe_cmd_err` interrupt is issued, and the flash controller clears the pipeline command queue. The flash controller services the violating incoming MAP01 read or write request with a normal page read or write sequence.

For a multi-plane device that supports multi-plane programming, you must set the `flag` bit of the `multiplane_operation` register in the `config` group to 1. In this case, the data is interleaved into page-size chunks to consecutive blocks.

A `pipe_cpyback_cmd_comp` interrupt is generated when the NAND flash controller has finished processing a pipeline command and has discarded that command from its queue. At this point of time, the

host can send another pipeline command. A pipeline command is popped from the queue, and an interrupt is issued when the flash controller has started processing the last page of pipeline command. Hence, the `pipe_cpyback_cmd_comp` interrupt is issued prior to the last page load in the case of a pipeline read command and start of data transfer of the last page to be programmed, in the case of a pipeline write command.

An additional `program_comp` interrupt is generated when the last page program operation completes in the case of a pipeline write command.

If the device command set requires the NAND flash controller to issue a load command for the last page in the pipeline read command, a `load_comp` interrupt is generated after the last page load operation completes.

The pipeline commands sequence advanced commands in the device, such as cache and multi-plane. When the NAND flash controller receives a multi-page read or write pipeline command, it sequences commands sent to the device depending on settings in the following registers in the `config` group:

- `cache_read_enable`
- `cache_write_enable`
- `multiplane_operation`

For a device that supports cache read sequences, the `flag` bit of the `cache_read_enable` register must be set to 1. The NAND flash controller sequences each multi-page pipeline read command as a cache read sequence. For a device that supports cache program command sequences, `cache_write_enable` must be set. The flash controller sequences each multi-page write pipeline command as a cache write sequence.

For a device that has multi-planes and supports multi-plane program commands, the NAND flash controller register `multiplane_operation`, in the `config` group, must be set. On receiving the multi-page pipeline write command, the flash controller sequences the device with multi-plane program commands and expects that the host transfers data to the flash controller in an even-odd block increment addressing mode.

NAND Flash Controller Address Map and Register Definitions

For complete HPS address map and register definitions, refer to the **Arria 10 HPS Address Map and Register Definitions**.

Related Information

Error Checking and Correction Controller on page 11-1
For more information about how to program the ECC registers, refer to this chapter.

nandecc_ecc Address Map

Module Instance	Base Address	End Address
`ecc_nandecc_ecc_registerBlock`	`0xFF8C2000`	`0xFF8C23FF`

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
IP_REV_ID on page 11-117	0x0	32	RO	0x0	
CTRL on page 11-117	0x8	32	RW	0x0	ECC Control Register
INITSTAT on page 11-118	0xC	32	RW	0x0	This bit is used to set the initialize the memory and ecc to a known value
ERRINTEN on page 11-119	0x10	32	RW	0x0	Error Interrupt enable
ERRINTENS on page 11-120	0x14	32	RW	0x0	Error Interrupt set
ERRINTENR on page 11-120	0x18	32	RW	0x0	Error Interrupt reset.
INTMODE on page 11-121	0x1C	32	RW	0x0	Reads reflect SERRINTEN.
INTSTAT on page 11-122	0x20	32	RW	0x0	This bit is used to enable interrupt generation on SERR lookup table overflow. When all the entries in the table are valid=1 and this is bit is enabled, serr_req signal will be asserted.
INTTEST on page 11-123	0x24	32	RW	0x0	This bits is used to test interrupt from ECC RAM to GIC
MODSTAT on page 11-124	0x28	32	RW	0x0	Counter feature status flag
DERRADDRA on page 11-124	0x2C	32	RO	0x0	This register shows the address of PORTA current double-bit error. RAM size will determine the maximum number of address bits.

Register	Offset	Width	Access	Reset Value	Description
SERRADDRA on page 11-125	0x30	32	RO	0x0	This register shows the address of PORTA current single-bit error. RAM size will determine the maximum number of address bits.
SERRCNTREG on page 11-126	0x3C	32	RW	0x0	Maximum counter value for single-bit error interrupt
ECC_Addrbus on page 11-127	0x40	32	RW	0x0	MSB bit of address is determined by ADR.
ECC_RData0bus on page 11-127	0x44	32	RO	0x0	Data will be read to this register field.
ECC_RData1bus on page 11-128	0x48	32	RO	0x0	Data will be read to this register field.
ECC_RData2bus on page 11-129	0x4C	32	RO	0x0	Data will be read to this register field.
ECC_RData3bus on page 11-129	0x50	32	RO	0x0	Data will be read to this register field.
ECC_WData0bus on page 11-130	0x54	32	WO	0x0	Data from the register will be written to the RAM.
ECC_WData1bus on page 11-131	0x58	32	WO	0x0	Data from the register will be written to the RAM.
ECC_WData2bus on page 11-131	0x5C	32	WO	0x0	Data from the register will be written to the RAM.
ECC_WData3bus on page 11-132	0x60	32	WO	0x0	Data from the register will be written to the RAM.
ECC_RDataecc0bus on page 11-133	0x64	32	RO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Register	Offset	Width	Access	Reset Value	Description
ECC_RDataecc1bus on page 11-133	0x68	32	RO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.
ECC_WDataecc0bus on page 11-134	0x6C	32	WO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.
ECC_WDataecc1bus on page 11-135	0x70	32	WO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.
ECC_dbytectrl on page 11-136	0x74	32	RW	0x0	Max number of implemented byte enabled is DAT/8
ECC_accctrl on page 11-137	0x78	32	RW	0x0	These bits determine which byte of data/ecc to write to RAM.
ECC_startacc on page 11-138	0x7C	32	RW	0x0	These bits determine which byte of data/ecc to write to RAM.
ECC_wdctrl on page 11-139	0x80	32	RW	0x0	Bits to Enable/Disable Watch Dog Timer
SERRLKUPA0 on page 11-139	0x90	32	RW	0x0	Single-bit error address in LOOKUP TABLE for PORTA.

nandecc_ecc Summary

Base Address: 0xFF8C2000

Register Address Offset	Bit Fields															
ecc_nandecc_ecc_register-Block																
IP_REV_ID 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SIREV RO 0x0															
CTRL 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															INITA 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							CNT_RSTA 0x0	Reserved							ECC_EN 0x0
INITSTAT 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															INITCOM-PLETEA 0x0
ERRINTEN 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTEN 0x0
ERRINTENS 0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTS 0x0
ERRINTENR 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTR 0x0

Register Address Offset	Bit Fields															
INTMODE 0x1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															INTONCMP 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							INTONOVF 0x0	Reserved							INTMODE 0x0
INTSTAT 0x20	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							DERRPENA 0x0	Reserved							SERRPENA 0x0
INTTEST 0x24	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							TDERRA 0x0	Reserved							TSERRA 0x0
MODSTAT 0x28	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															CMPFLGA 0x0
DERRADDRA 0x2C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						Address 0x0									
SERRADDRA 0x30	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						Address 0x0									

Register Address Offset	Bit Fields															
SERRCNTREG 0x3C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	SERRCNT 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SERRCNT 0x0															
ECC_Addrbus 0x40	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						ECC_AddrBUS 0x0									
ECC_RData0bus 0x44	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_RData1bus 0x48	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_RData2bus 0x4C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_RData3bus 0x50	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															

Register Address Offset	Bit Fields															
ECC_WData0bus 0x54	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_WData1bus 0x58	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_WData2bus 0x5C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_WData3bus 0x60	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_RDataecc0bus 0x64	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved		ECC_RDataecc3BUS 0x0						Reserved		ECC_RDataecc2BUS 0x0					
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		ECC_RDataecc1BUS 0x0						Reserved		ECC_RDataecc0BUS 0x0					
ECC_RDataecc1bus 0x68	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved		ECC_RDataecc7BUS 0x0						Reserved		ECC_RDataecc6BUS 0x0					
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		ECC_RDataecc5BUS 0x0						Reserved		ECC_RDataecc4BUS 0x0					

Register Address Offset	Bit Fields															
ECC_WDataecc0bus 0x6C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved		ECC_WDataecc3BUS 0x0						Reserved		ECC_WDataecc2BUS 0x0					
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		ECC_WDataecc1BUS 0x0						Reserved		ECC_WDataecc0BUS 0x0					
ECC_WDataecc1bus 0x70	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved		ECC_WDataecc7BUS 0x0						Reserved		ECC_WDataecc6BUS 0x0					
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		ECC_WDataecc5BUS 0x0						Reserved		ECC_WDataecc4BUS 0x0					
ECC_dbytectrl 0x74	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															DBEN 0x0
ECC_accctrl 0x78	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							RDWR 0x0	Reserved						ECCOVR 0x0	DATAOVR 0x0
ECC_startacc 0x7C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															ENBUSA 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															
ECC_wdctrl 0x80	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															WDEN_RAM 0x0

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SERRLKUPA0 0x90	VALID RW 0x0	Reserved														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						Address RO 0x0									

IP_REV_ID

Module Instance	Base Address	Register Address
ecc_nandecc_ecc_registerBlock	0xFF8C2000	0xFF8C2000

Offset: 0x0

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SIREV RO 0x0															

IP_REV_ID Fields

Bit	Name	Description	Access	Reset
15:0	SIREV	IP Rev # These bits indicate the silicon revision number.	RO	0x0

CTRL

ECC Control Register

Module Instance	Base Address	Register Address
ecc_nandecc_ecc_registerBlock	0xFF8C2000	0xFF8C2008

Offset: 0x8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															INITA 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							CNT_RSTA 0x0	Reserved							ECC_EN 0x0

CTRL Fields

Bit	Name	Description	Access	Reset
16	INITA	Enable for the hardware memory initialization PORTA.	RW	0x0
8	CNT_RSTA	Enable to reset internal single-bit error counter A value to zero	RW	0x0
0	ECC_EN	Enable for the ECC detection and correction logic.	RW	0x0

INITSTAT

This bit is used to set the initialize the memory and ecc to a known value

Module Instance	Base Address	Register Address
ecc_nandecc_ecc_registerBlock	0xFF8C2000	0xFF8C200C

Offset: 0xC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															INITCOM-PLETEA 0x0

INITSTAT Fields

Bit	Name	Description	Access	Reset
0	INITCOMPLETEA	This bit is used to verify if the hardware memory initialization has completed PORTB.	RW	0x0

ERRINTEN

Error Interrupt enable

Module Instance	Base Address	Register Address
ecc_nandecc_ecc_registerBlock	0xFF8C2000	0xFF8C2010

Offset: 0x10

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															SERRINTEN 0x0

ERRINTEN Fields

Bit	Name	Description	Access	Reset
0	SERRINTEN	This bit is used to enable the single bit error interrupt of ECC RAM system	RW	0x0

ERRINTENS

Error Interrupt set

Module Instance	Base Address	Register Address
ecc_nandecc_ecc_registerBlock	0xFF8C2000	0xFF8C2014

Offset: 0x14

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															SERRINTS 0x0

ERRINTENS Fields

Bit	Name	Description	Access	Reset
0	SERRINTS	This bit is used to set the single-bit error interrupt bit.	RW	0x0

ERRINTENR

Error Interrupt reset.

Module Instance	Base Address	Register Address
ecc_nandecc_ecc_registerBlock	0xFF8C2000	0xFF8C2018

Offset: 0x18

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															SERRINTR 0x0

ERRINTENR Fields

Bit	Name	Description	Access	Reset
0	SERRINTR	This bit is used to reset the single-bit error interrupt bit. Reads reflect SERRINTEN. 1'b0: Writing of zero has no effect. 1'b1: By writing one, this bit will reset SERRINTEN bit to 0. This is performing a bitwise writing of this feature.	RW	0x0

INTMODE

Reads reflect SERRINTEN.

Module Instance	Base Address	Register Address
ecc_nandecc_ecc_registerBlock	0xFF8C2000	0xFF8C201C

Offset: 0x1C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															INTONCMP 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							INTONOVF 0x0	Reserved							INTMODE 0x0

INTMODE Fields

Bit	Name	Description	Access	Reset
16	INTONCMP	Enable interrupt on compare.	RW	0x0
8	INTONOVF	Enable interrupt on overflow.	RW	0x0
0	INTMODE	Interrupt mode for single-bit errors.	RW	0x0

INTSTAT

This bit is used to enable interrupt generation on SERR lookup table overflow. When all the entries in the table are valid=1 and this is bit is enabled, serr_req signal will be asserted.

Module Instance	Base Address	Register Address
ecc_nandecc_ecc_registerBlock	0xFF8C2000	0xFF8C2020

Offset: 0x20

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							DERRPENA 0x0	Reserved							SERRPENA 0x0

INTSTAT Fields

Bit	Name	Description	Access	Reset
8	DERRPENA	Double-bit error pending for PORTA.	RW	0x0
0	SERRPENA	Single-bit error pending for PORTA.	RW	0x0

INTTEST

This bits is used to test interrupt from ECC RAM to GIC

Module Instance	Base Address	Register Address
ecc_nandecc_ecc_registerBlock	0xFF8C2000	0xFF8C2024

Offset: 0x24

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							TDERRA 0x0	Reserved							TSERRA 0x0

INTTEST Fields

Bit	Name	Description	Access	Reset
8	TDERRA	Test PORTA Double-bit error.	RW	0x0
0	TSERRA	Test PORTA Single-bit error.	RW	0x0

MODSTAT

Counter feature status flag

Module Instance	Base Address	Register Address
ecc_nandecc_ecc_registerBlock	0xFF8C2000	0xFF8C2028

Offset: 0x28

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															CMPFLGA 0x0

MODSTAT Fields

Bit	Name	Description	Access	Reset
0	CMPFLGA	Port A compare status flag	RW	0x0

DERRADDRA

This register shows the address of PORTA current double-bit error. RAM size will determine the maximum number of address bits.

Module Instance	Base Address	Register Address
ecc_nandecc_ecc_registerBlock	0xFF8C2000	0xFF8C202C

Offset: 0x2C

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						Address 0x0									

DERRADDRA Fields

Bit	Name	Description	Access	Reset
9:0	Address	Recent double-bit error address.	RO	0x0

SERRADDRA

This register shows the address of PORTA current single-bit error. RAM size will determine the maximum number of address bits.

Module Instance	Base Address	Register Address
ecc_nandecc_ecc_registerBlock	0xFF8C2000	0xFF8C2030

Offset: 0x30

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						Address 0x0									

SERRADDRA Fields

Bit	Name	Description	Access	Reset
9:0	Address	Recent single-bit error address.	RO	0x0

SERRCNTREG

Maximum counter value for single-bit error interrupt

Module Instance	Base Address	Register Address
ecc_nandecc_ecc_registerBlock	0xFF8C2000	0xFF8C203C

Offset: 0x3C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SERRCNT 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SERRCNT 0x0															

SERRCNTREG Fields

Bit	Name	Description	Access	Reset
31:0	SERRCNT	Counter value	RW	0x0

ECC_Addrbus

MSB bit of address is determined by ADR.

Module Instance	Base Address	Register Address
ecc_nandecc_ecc_registerBlock	0xFF8C2000	0xFF8C2040

Offset: 0x40

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						ECC_AddrBUS 0x0									

ECC_Addrbus Fields

Bit	Name	Description	Access	Reset
9:0	ECC_AddrBUS	Address will be driven to RAM to either read or write the data. Address will be latched by the RAM when the Enbus is asserted.	RW	0x0

ECC_RData0bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_nandecc_ecc_registerBlock	0xFF8C2000	0xFF8C2044

Offset: 0x44

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData0bus Fields

Bit	Name	Description	Access	Reset
15:0	ECC_RDataBUS	ECC_RDataBUS[31:0].	RO	0x0

ECC_RData1bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_nandecc_ecc_registerBlock	0xFF8C2000	0xFF8C2048

Offset: 0x48

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData1bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_RDataBUS	ECC_RDataBUS[63:32].	RO	0x0

ECC_RData2bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_nandecc_ecc_registerBlock	0xFF8C2000	0xFF8C204C

Offset: 0x4C

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData2bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_RDataBUS	ECC_RDataBUS[95:64].	RO	0x0

ECC_RData3bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_nandecc_ecc_registerBlock	0xFF8C2000	0xFF8C2050

Offset: 0x50

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData3bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_RDataBUS	ECC_RDataBUS[127-96].	RO	0x0

ECC_WData0bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_nandecc_ecc_registerBlock	0xFF8C2000	0xFF8C2054

Offset: 0x54

Access: WO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData0bus Fields

Bit	Name	Description	Access	Reset
15:0	ECC_WDataBUS	ECC_WDataBUS[31:0].	WO	0x0

ECC_WData1bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_nandecc_ecc_registerBlock	0xFF8C2000	0xFF8C2058

Offset: 0x58

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData1bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_WDataBUS	ECC_WDataBUS[63:32].	WO	0x0

ECC_WData2bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_nandecc_ecc_registerBlock	0xFF8C2000	0xFF8C205C

Offset: 0x5C

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData2bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_WDataBUS	ECC_WDataBUS[95-64].	WO	0x0

ECC_WData3bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_nandecc_ecc_registerBlock	0xFF8C2000	0xFF8C2060

Offset: 0x60

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData3bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_WDataBUS	ECC_WDataBUS[127-96].	WO	0x0

ECC_RDataecc0bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_nandecc_ecc_registerBlock	0xFF8C2000	0xFF8C2064

Offset: 0x64

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		ECC_RDataecc3BUS 0x0						Reserved		ECC_RDataecc2BUS 0x0					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		ECC_RDataecc1BUS 0x0						Reserved		ECC_RDataecc0BUS 0x0					

ECC_RDataecc0bus Fields

Bit	Name	Description	Access	Reset
29:24	ECC_RDataecc3BUS	Eccdata will be read to this register field.	RO	0x0
21:16	ECC_RDataecc2BUS	Eccdata will be read to this register field.	RO	0x0
13:8	ECC_RDataecc1BUS	Eccdata will be read to this register field.	RO	0x0
5:0	ECC_RDataecc0BUS	Eccdata will be read to this register field.	RO	0x0

ECC_RDataecc1bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_nandecc_ecc_registerBlock	0xFF8C2000	0xFF8C2068

Offset: 0x68

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		ECC_RDataecc7BUS 0x0						Reserved		ECC_RDataecc6BUS 0x0					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		ECC_RDataecc5BUS 0x0						Reserved		ECC_RDataecc4BUS 0x0					

ECC_RDataecc1bus Fields

Bit	Name	Description	Access	Reset
29:24	ECC_RDataecc7BUS	Eccdata will be read to this register field.	RO	0x0
21:16	ECC_RDataecc6BUS	Eccdata will be read to this register field.	RO	0x0
13:8	ECC_RDataecc5BUS	Eccdata will be read to this register field.	RO	0x0
5:0	ECC_RDataecc4BUS	Eccdata will be read to this register field.	RO	0x0

ECC_WDataecc0bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_nandecc_ecc_registerBlock	0xFF8C2000	0xFF8C206C

Offset: 0x6C

Access: WO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		ECC_WDataecc3BUS 0x0						Reserved		ECC_WDataecc2BUS 0x0					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		ECC_WDataecc1BUS 0x0						Reserved		ECC_WDataecc0BUS 0x0					

ECC_WDataecc0bus Fields

Bit	Name	Description	Access	Reset
29:24	ECC_WDataecc3BUS	Eccdata from the register will be written to the RAM.	WO	0x0
21:16	ECC_WDataecc2BUS	Eccdata from the register will be written to the RAM.	WO	0x0
13:8	ECC_WDataecc1BUS	Eccdata from the register will be written to the RAM.	WO	0x0
5:0	ECC_WDataecc0BUS	Eccdata from the register will be written to the RAM.	WO	0x0

ECC_WDataecc1bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_nandecc_ecc_registerBlock	0xFF8C2000	0xFF8C2070

Offset: 0x70

Access: WO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		ECC_WDataecc7BUS 0x0						Reserved		ECC_WDataecc6BUS 0x0					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		ECC_WDataecc5BUS 0x0						Reserved		ECC_WDataecc4BUS 0x0					

ECC_WDataecc1bus Fields

Bit	Name	Description	Access	Reset
29:24	ECC_WDataecc7BUS	Eccdata from the register will be written to the RAM.	WO	0x0
21:16	ECC_WDataecc6BUS	Eccdata from the register will be written to the RAM.	WO	0x0
13:8	ECC_WDataecc5BUS	Eccdata from the register will be written to the RAM.	WO	0x0
5:0	ECC_WDataecc4BUS	Eccdata from the register will be written to the RAM.	WO	0x0

ECC_dbytectrl

ECC Data Byte Control. The maximum number of implemented byte enables equals the data width of the RAM divided by 8.

Module Instance	Base Address	Register Address
ecc_nandecc_ecc_registerBlock	0xFF8C2000	0xFF8C2074

Offset: 0x74

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															DBEN 0x0

ECC_dbytectrl Fields

Bit	Name	Description	Access	Reset
0	DBEN	Byte or word enable for access.	RW	0x0

ECC_accctrl

These bits determine which byte of data/ecc to write to RAM.

Module Instance	Base Address	Register Address
ecc_nandecc_ecc_registerBlock	0xFF8C2000	0xFF8C2078

Offset: 0x78

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							RDWR 0x0	Reserved						ECCOVR 0x0	DATAOVR 0x0

ECC_accctrl Fields

Bit	Name	Description	Access	Reset
8	RDWR	Control for read/write.	RW	0x0
1	ECCOVR	ECC Data Override.	RW	0x0
0	DATAOVR	RAM Data Override. Override the ECC_dataBUS register with RAM data in read mode set by ECC_RW. 1'b0: Data override disabled. 1'b1: Data override enabled.	RW	0x0

ECC_startacc

These bits determine which byte of data/ecc to write to RAM.

Module Instance	Base Address	Register Address
ecc_nandecc_ecc_registerBlock	0xFF8C2000	0xFF8C207C

Offset: 0x7C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															ENBUSA 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															

ECC_startacc Fields

Bit	Name	Description	Access	Reset
16	ENBUSA	Start RAM access for PORTA.	RW	0x0

ECC_wdctrl

Bits to Enable/Disable Watch Dog Timer

Module Instance	Base Address	Register Address
ecc_nandecc_ecc_registerBlock	0xFF8C2000	0xFF8C2080

Offset: 0x80

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															WDEN_RAM 0x0

ECC_wdctrl Fields

Bit	Name	Description	Access	Reset
0	WDEN_RAM	Enable watchdog timeout for OCP register access to IP RAM.	RW	0x0

SERRLKUPA0

Single-bit error address in LOOKUP TABLE for PORTA.

Module Instance	Base Address	Register Address
ecc_nandecc_ecc_registerBlock	0xFF8C2000	0xFF8C2090

Offset: 0x90

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VALID RW 0x0	Reserved														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						Address RO 0x0									

SERRLKUPA0 Fields

Bit	Name	Description	Access	Reset
31	VALID	Valid flag bit. Valid bit indicates if the address in this register is current or stale.	RW	0x0
9:0	Address	Recent Single-bit error address. This register shows the address of the each single-bit error. RAM size will determine the maximum number of address bits. If ram size is 32 Kbytes, bit 30-16 will be reserved and read as zero.	RO	0x0

nandr_ecc Address Map

Module Instance	Base Address	End Address
ecc_nandr_ecc_registerBlock	0xFF8C2400	0xFF8C27FF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
IP_REV_ID on page 11-149	0x0	32	RO	0x0	
CTRL on page 11-150	0x8	32	RW	0x0	ECC Control Register
INITSTAT on page 11-150	0xC	32	RW	0x0	This bit is used to set the initialize the memory and ecc to a known value
ERRINTEN on page 11-151	0x10	32	RW	0x0	Error Interrupt enable
ERRINTENS on page 11-152	0x14	32	RW	0x0	Error Interrupt set
ERRINTENR on page 11-153	0x18	32	RW	0x0	Error Interrupt reset.

Register	Offset	Width	Access	Reset Value	Description
INTMODE on page 11-154	0x1C	32	RW	0x0	Reads reflect SERRINTEN.
INTSTAT on page 11-154	0x20	32	RW	0x0	This bit is used to enable interrupt generation on SERR lookup table overflow. When all the entries in the table are valid=1 and this is bit is enabled, serr_req signal will be asserted.
INTTEST on page 11-155	0x24	32	RW	0x0	This bits is used to test interrupt from ECC RAM to GIC
MODSTAT on page 11-156	0x28	32	RW	0x0	Counter feature status flag
DERRADDRA on page 11-157	0x2C	32	RO	0x0	This register shows the address of PORTA current double-bit error. RAM size will determine the maximum number of address bits.
SERRADDRA on page 11-157	0x30	32	RO	0x0	This register shows the address of PORTA current single-bit error. RAM size will determine the maximum number of address bits.
SERRCNTREG on page 11-158	0x3C	32	RW	0x0	Maximum counter value for single-bit error interrupt
ECC_Addrbus on page 11-159	0x40	32	RW	0x0	MSB bit of address is determined by ADR.
ECC_RData0bus on page 11-160	0x44	32	RO	0x0	Data will be read to this register field.
ECC_RData1bus on page 11-160	0x48	32	RO	0x0	Data will be read to this register field.
ECC_RData2bus on page 11-161	0x4C	32	RO	0x0	Data will be read to this register field.

Register	Offset	Width	Access	Reset Value	Description
ECC_RData3bus on page 11-162	0x50	32	RO	0x0	Data will be read to this register field.
ECC_WData0bus on page 11-162	0x54	32	WO	0x0	Data from the register will be written to the RAM.
ECC_WData1bus on page 11-163	0x58	32	WO	0x0	Data from the register will be written to the RAM.
ECC_WData2bus on page 11-164	0x5C	32	WO	0x0	Data from the register will be written to the RAM.
ECC_WData3bus on page 11-164	0x60	32	WO	0x0	Data from the register will be written to the RAM.
ECC_RDataecc0bus on page 11-165	0x64	32	RO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.
ECC_RDataecc1bus on page 11-166	0x68	32	RO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.
ECC_WDataecc0bus on page 11-167	0x6C	32	WO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.
ECC_WDataecc1bus on page 11-168	0x70	32	WO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Register	Offset	Width	Access	Reset Value	Description
ECC_dbytectrl on page 11-169	0x74	32	RW	0x0	Max number of implemented byte enabled is DAT/8
ECC_accctrl on page 11-170	0x78	32	RW	0x0	These bits determine which byte of data/ecc to write to RAM.
ECC_startacc on page 11-170	0x7C	32	RW	0x0	These bits determine which byte of data/ecc to write to RAM.
ECC_wdctrl on page 11-171	0x80	32	RW	0x0	Bits to Enable/Disable Watch Dog Timer
SERRLKUPA0 on page 11-172	0x90	32	RW	0x0	Single-bit error address in LOOKUP TABLE for PORTA.

nandr_ecc Summary

Base Address: 0xFF8C2400

Register Address Offset	Bit Fields															
ecc_nandr_ecc_register-Block																
IP_REV_ID 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SIREV RO 0x0															
CTRL 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															INITA 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							CNT_RSTA 0x0	Reserved							ECC_EN 0x0

Register Address Offset	Bit Fields															
INITSTAT 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															INITCOM-PLETEA 0x0
ERRINTEN 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTEN 0x0
ERRINTENS 0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTS 0x0
ERRINTENR 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTR 0x0
INTMODE 0x1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															INTONCMP 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							INTONOVF 0x0	Reserved							INTMODE 0x0
INTSTAT 0x20	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							DERRPENA 0x0	Reserved							SERRPENA 0x0

Register Address Offset	Bit Fields															
INTTEST 0x24	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							TDERRA 0x0	Reserved							TSERRA 0x0
MODSTAT 0x28	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															CMPFLGA 0x0
DERRADDRA 0x2C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved											Address 0x0				
SERRADDRA 0x30	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved											Address 0x0				
SERRCNTREG 0x3C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	SERRCNT 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SERRCNT 0x0															
ECC_Addrbus 0x40	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved											ECC_AddrBUS 0x0				

Register Address Offset	Bit Fields															
ECC_RData0bus 0x44	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_RData1bus 0x48	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_RData2bus 0x4C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_RData3bus 0x50	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_WData0bus 0x54	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_WData1bus 0x58	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															

Register Address Offset	Bit Fields															
ECC_WData2bus 0x5C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_WData3bus 0x60	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_RDataec0bus 0x64	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_RDataecc3BUS 0x0							Reserved	ECC_RDataecc2BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_RDataecc1BUS 0x0							Reserved	ECC_RDataecc0BUS 0x0						
ECC_RDataec1bus 0x68	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_RDataecc7BUS 0x0							Reserved	ECC_RDataecc6BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_RDataecc5BUS 0x0							Reserved	ECC_RDataecc4BUS 0x0						
ECC_WDataec0bus 0x6C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_WDataecc3BUS 0x0							Reserved	ECC_WDataecc2BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_WDataecc1BUS 0x0							Reserved	ECC_WDataecc0BUS 0x0						
ECC_WDataec1bus 0x70	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_WDataecc7BUS 0x0							Reserved	ECC_WDataecc6BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_WDataecc5BUS 0x0							Reserved	ECC_WDataecc4BUS 0x0						

Register Address Offset	Bit Fields															
ECC_dbytectrl 0x74	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															DBEN 0x0
ECC_accctrl 0x78	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							RDWR 0x0	Reserved						ECCOVR 0x0	DATAOVR 0x0
ECC_startacc 0x7C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															ENBUSA 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															
ECC_wdctrl 0x80	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															WDEN_RAM 0x0
SERRLKUPA0 0x90	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	VALID RW 0x0	Reserved														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved											Address RO 0x0				

IP_REV_ID

Module Instance	Base Address	Register Address
ecc_nandr_ecc_registerBlock	0xFF8C2400	0xFF8C2400

Offset: 0x0

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SIREV RO 0x0															

IP_REV_ID Fields

Bit	Name	Description	Access	Reset
15:0	SIREV	IP Rev # These bits indicate the silicon revision number.	RO	0x0

CTRL

ECC Control Register

Module Instance	Base Address	Register Address
ecc_nandr_ecc_registerBlock	0xFF8C2400	0xFF8C2408

Offset: 0x8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															INITA 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							CNT_RSTA 0x0	Reserved							ECC_EN 0x0

CTRL Fields

Bit	Name	Description	Access	Reset
16	INITA	Enable for the hardware memory initialization PORTA.	RW	0x0
8	CNT_RSTA	Enable to reset internal single-bit error counter A value to zero	RW	0x0
0	ECC_EN	Enable for the ECC detection and correction logic.	RW	0x0

INITSTAT

This bit is used to set the initialize the memory and ecc to a known value

Module Instance	Base Address	Register Address
ecc_nandr_ecc_registerBlock	0xFF8C2400	0xFF8C240C

Offset: 0xC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															INITCOM-PLETEA 0x0

INITSTAT Fields

Bit	Name	Description	Access	Reset
0	INITCOMPLETEA	This bit is used to verify if the hardware memory initialization has completed PORTB.	RW	0x0

ERRINTEN

Error Interrupt enable

Module Instance	Base Address	Register Address
ecc_nandr_ecc_registerBlock	0xFF8C2400	0xFF8C2410

Offset: 0x10

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															SERRINTEN 0x0

ERRINTEN Fields

Bit	Name	Description	Access	Reset
0	SERRINTEN	This bit is used to enable the single bit error interrupt of ECC RAM system	RW	0x0

ERRINTENS

Error Interrupt set

Module Instance	Base Address	Register Address
ecc_nandr_ecc_registerBlock	0xFF8C2400	0xFF8C2414

Offset: 0x14

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															SERRINTS 0x0

ERRINTENS Fields

Bit	Name	Description	Access	Reset
0	SERRINTS	This bit is used to set the single-bit error interrupt bit.	RW	0x0

ERRINTENR

Error Interrupt reset.

Module Instance	Base Address	Register Address
ecc_nandr_ecc_registerBlock	0xFF8C2400	0xFF8C2418

Offset: 0x18

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															SERRINTR 0x0

ERRINTENR Fields

Bit	Name	Description	Access	Reset
0	SERRINTR	This bit is used to reset the single-bit error interrupt bit. Reads reflect SERRINTEN. 1'b0: Writing of zero has no effect. 1'b1: By writing one, this bit will reset SERRINTEN bit to 0. This is performing a bitwise writing of this feature.	RW	0x0

INTMODE

Reads reflect SERRINTEN.

Module Instance	Base Address	Register Address
ecc_nandr_ecc_registerBlock	0xFF8C2400	0xFF8C241C

Offset: 0x1C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															INTONCMP 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							INTONOVF 0x0	Reserved							INTMODE 0x0

INTMODE Fields

Bit	Name	Description	Access	Reset
16	INTONCMP	Enable interrupt on compare.	RW	0x0
8	INTONOVF	Enable interrupt on overflow.	RW	0x0
0	INTMODE	Interrupt mode for single-bit errors.	RW	0x0

INTSTAT

This bit is used to enable interrupt generation on SERR lookup table overflow. When all the entries in the table are valid=1 and this is bit is enabled, serr_req signal will be asserted.

Module Instance	Base Address	Register Address
ecc_nandr_ecc_registerBlock	0xFF8C2400	0xFF8C2420

Offset: 0x20

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							DERRPENA 0x0	Reserved							SERRPENA 0x0

INTSTAT Fields

Bit	Name	Description	Access	Reset
8	DERRPENA	Double-bit error pending for PORTA.	RW	0x0
0	SERRPENA	Single-bit error pending for PORTA.	RW	0x0

INTTEST

This bits is used to test interrupt from ECC RAM to GIC

Module Instance	Base Address	Register Address
ecc_nandr_ecc_registerBlock	0xFF8C2400	0xFF8C2424

Offset: 0x24

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							TDERRA 0x0	Reserved							TSERRA 0x0

INTTEST Fields

Bit	Name	Description	Access	Reset
8	TDERRA	Test PORTA Double-bit error.	RW	0x0
0	TSERRA	Test PORTA Single-bit error.	RW	0x0

MODSTAT

Counter feature status flag

Module Instance	Base Address	Register Address
ecc_nandr_ecc_registerBlock	0xFF8C2400	0xFF8C2428

Offset: 0x28

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															CMPFLGA 0x0

MODSTAT Fields

Bit	Name	Description	Access	Reset
0	CMPFLGA	Port A compare status flag	RW	0x0

DERRADDRA

This register shows the address of PORTA current double-bit error. RAM size will determine the maximum number of address bits.

Module Instance	Base Address	Register Address
ecc_nandr_ecc_registerBlock	0xFF8C2400	0xFF8C242C

Offset: 0x2C

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved											Address 0x0				

DERRADDRA Fields

Bit	Name	Description	Access	Reset
4:0	Address	Recent double-bit error address.	RO	0x0

SERRADDRA

This register shows the address of PORTA current single-bit error. RAM size will determine the maximum number of address bits.

Module Instance	Base Address	Register Address
ecc_nandr_ecc_registerBlock	0xFF8C2400	0xFF8C2430

Offset: 0x30

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved											Address 0x0				

SERRADDRA Fields

Bit	Name	Description	Access	Reset
4:0	Address	Recent single-bit error address.	RO	0x0

SERRCNTREG

Maximum counter value for single-bit error interrupt

Module Instance	Base Address	Register Address
ecc_nandr_ecc_registerBlock	0xFF8C2400	0xFF8C243C

Offset: 0x3C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SERRCNT 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SERRCNT 0x0															

SERRCNTREG Fields

Bit	Name	Description	Access	Reset
31:0	SERRCNT	Counter value	RW	0x0

ECC_Addrbus

MSB bit of address is determined by ADR.

Module Instance	Base Address	Register Address
ecc_nandr_ecc_registerBlock	0xFF8C2400	0xFF8C2440

Offset: 0x40

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved											ECC_AddrBUS 0x0				

ECC_Addrbus Fields

Bit	Name	Description	Access	Reset
4:0	ECC_AddrBUS	Address will be driven to RAM to either read or write the data. Address will be latched by the RAM when the Enbus is asserted.	RW	0x0

ECC_RData0bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_nandr_ecc_registerBlock	0xFF8C2400	0xFF8C2444

Offset: 0x44

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData0bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_RDataBUS	ECC_RDataBUS[31:0].	RO	0x0

ECC_RData1bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_nandr_ecc_registerBlock	0xFF8C2400	0xFF8C2448

Offset: 0x48

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData1bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_RDataBUS	ECC_RDataBUS[63:32].	RO	0x0

ECC_RData2bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_nandr_ecc_registerBlock	0xFF8C2400	0xFF8C244C

Offset: 0x4C

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData2bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_RDataBUS	ECC_RDataBUS[95:64].	RO	0x0

ECC_RData3bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_nandr_ecc_registerBlock	0xFF8C2400	0xFF8C2450

Offset: 0x50

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData3bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_RDataBUS	ECC_RDataBUS[127-96].	RO	0x0

ECC_WData0bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_nandr_ecc_registerBlock	0xFF8C2400	0xFF8C2454

Offset: 0x54

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData0bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_WDataBUS	ECC_WDataBUS[31:0].	WO	0x0

ECC_WData1bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_nandr_ecc_registerBlock	0xFF8C2400	0xFF8C2458

Offset: 0x58

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData1bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_WDataBUS	ECC_WDataBUS[63:32].	WO	0x0

ECC_WData2bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_nandr_ecc_registerBlock	0xFF8C2400	0xFF8C245C

Offset: 0x5C

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData2bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_WDataBUS	ECC_WDataBUS[95-64].	WO	0x0

ECC_WData3bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_nandr_ecc_registerBlock	0xFF8C2400	0xFF8C2460

Offset: 0x60

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData3bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_WDataBUS	ECC_WDataBUS[127-96].	WO	0x0

ECC_RDataecc0bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_nandr_ecc_registerBlock	0xFF8C2400	0xFF8C2464

Offset: 0x64

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	ECC_RDataecc3BUS 0x0							Reserved	ECC_RDataecc2BUS 0x0						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	ECC_RDataecc1BUS 0x0							Reserved	ECC_RDataecc0BUS 0x0						

ECC_RDataecc0bus Fields

Bit	Name	Description	Access	Reset
30:24	ECC_RDataecc3BUS	Eccdata will be read to this register field.	RO	0x0
22:16	ECC_RDataecc2BUS	Eccdata will be read to this register field.	RO	0x0
14:8	ECC_RDataecc1BUS	Eccdata will be read to this register field.	RO	0x0
6:0	ECC_RDataecc0BUS	Eccdata will be read to this register field.	RO	0x0

ECC_RDataecc1bus

The msb bit for the register is configured based on DAT

parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_nandr_ecc_registerBlock	0xFF8C2400	0xFF8C2468

Offset: 0x68

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	ECC_RDataecc7BUS 0x0							Reserved	ECC_RDataecc6BUS 0x0						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	ECC_RDataecc5BUS 0x0							Reserved	ECC_RDataecc4BUS 0x0						

ECC_RDataecc1bus Fields

Bit	Name	Description	Access	Reset
30:24	ECC_RDataecc7BUS	Eccdata will be read to this register field.	RO	0x0
22:16	ECC_RDataecc6BUS	Eccdata will be read to this register field.	RO	0x0
14:8	ECC_RDataecc5BUS	Eccdata will be read to this register field.	RO	0x0
6:0	ECC_RDataecc4BUS	Eccdata will be read to this register field.	RO	0x0

ECC_WDataecc0bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_nandr_ecc_registerBlock	0xFF8C2400	0xFF8C246C

Offset: 0x6C

Access: WO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	ECC_WDataecc3BUS 0x0							Reserved	ECC_WDataecc2BUS 0x0						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	ECC_WDataecc1BUS 0x0							Reserved	ECC_WDataecc0BUS 0x0						

ECC_WDataecc0bus Fields

Bit	Name	Description	Access	Reset
30:24	ECC_WDataecc3BUS	Eccdata from the register will be written to the RAM.	WO	0x0
22:16	ECC_WDataecc2BUS	Eccdata from the register will be written to the RAM.	WO	0x0
14:8	ECC_WDataecc1BUS	Eccdata from the register will be written to the RAM.	WO	0x0
6:0	ECC_WDataecc0BUS	Eccdata from the register will be written to the RAM.	WO	0x0

ECC_WDataecc1bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_nandr_ecc_registerBlock	0xFF8C2400	0xFF8C2470

Offset: 0x70

Access: WO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	ECC_WDataecc7BUS 0x0							Reserved	ECC_WDataecc6BUS 0x0						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	ECC_WDataecc5BUS 0x0							Reserved	ECC_WDataecc4BUS 0x0						

ECC_WDataecc1bus Fields

Bit	Name	Description	Access	Reset
30:24	ECC_WDataecc7BUS	Eccdata from the register will be written to the RAM.	WO	0x0
22:16	ECC_WDataecc6BUS	Eccdata from the register will be written to the RAM.	WO	0x0
14:8	ECC_WDataecc5BUS	Eccdata from the register will be written to the RAM.	WO	0x0
6:0	ECC_WDataecc4BUS	Eccdata from the register will be written to the RAM.	WO	0x0

ECC_dbytectrl

ECC Data Byte Control. The maximum number of implemented byte enables equals the data width of the RAM divided by 8.

Module Instance	Base Address	Register Address
ecc_nandr_ecc_registerBlock	0xFF8C2400	0xFF8C2474

Offset: 0x74

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															DBEN 0x0

ECC_dbytectrl Fields

Bit	Name	Description	Access	Reset
0	DBEN	Byte or word enable for access.	RW	0x0

ECC_accctrl

These bits determine which byte of data/ecc to write to RAM.

Module Instance	Base Address	Register Address
ecc_nandr_ecc_registerBlock	0xFF8C2400	0xFF8C2478

Offset: 0x78

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							RDWR 0x0	Reserved						ECCOVR 0x0	DATAOVR 0x0

ECC_accctrl Fields

Bit	Name	Description	Access	Reset
8	RDWR	Control for read/write.	RW	0x0
1	ECCOVR	ECC Data Override.	RW	0x0
0	DATAOVR	RAM Data Override. Override the ECC_dataBUS register with RAM data in read mode set by ECC_RW. 1'b0: Data override disabled. 1'b1: Data override enabled.	RW	0x0

ECC_startacc

These bits determine which byte of data/ecc to write to RAM.

Module Instance	Base Address	Register Address
ecc_nandr_ecc_registerBlock	0xFF8C2400	0xFF8C247C

Offset: 0x7C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															ENBUSA 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															

ECC_startacc Fields

Bit	Name	Description	Access	Reset
16	ENBUSA	Start RAM access for PORTA.	RW	0x0

ECC_wdctrl

Bits to Enable/Disable Watch Dog Timer

Module Instance	Base Address	Register Address
ecc_nandr_ecc_registerBlock	0xFF8C2400	0xFF8C2480

Offset: 0x80

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															WDEN_RAM 0x0

ECC_wdctrl Fields

Bit	Name	Description	Access	Reset
0	WDEN_RAM	Enable watchdog timeout for OCP register access to IP RAM.	RW	0x0

SERRLKUPA0

Single-bit error address in LOOKUP TABLE for PORTA.

Module Instance	Base Address	Register Address
ecc_nandr_ecc_registerBlock	0xFF8C2400	0xFF8C2490

Offset: 0x90

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VALID RW 0x0	Reserved														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved											Address RO 0x0				

SERRLKUPA0 Fields

Bit	Name	Description	Access	Reset
31	VALID	Valid flag bit. Valid bit indicates if the address in this register is current or stale.	RW	0x0
4:0	Address	Recent Single-bit error address. This register shows the address of the each single-bit error. RAM size will determine the maximum number of address bits. If ram size is 32 Kbytes, bit 30-16 will be reserved and read as zero.	RO	0x0

nandw_ecc Address Map

Module Instance	Base Address	End Address
ecc_nandw_ecc_registerBlock	0xFF8C2800	0xFF8C2BFF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
IP_REV_ID on page 11-181	0x0	32	RO	0x0	
CTRL on page 11-182	0x8	32	RW	0x0	ECC Control Register

Register	Offset	Width	Access	Reset Value	Description
INITSTAT on page 11-183	0xC	32	RW	0x0	This bit is used to set the initialize the memory and ecc to a known value
ERRINTEN on page 11-184	0x10	32	RW	0x0	Error Interrupt enable
ERRINTENS on page 11-185	0x14	32	RW	0x0	Error Interrupt set
ERRINTENR on page 11-185	0x18	32	RW	0x0	Error Interrupt reset.
INTMODE on page 11-186	0x1C	32	RW	0x0	Reads reflect SERRINTEN.
INTSTAT on page 11-187	0x20	32	RW	0x0	This bit is used to enable interrupt generation on SERR lookup table overflow. When all the entries in the table are valid=1 and this is bit is enabled, serr_req signal will be asserted.
INTTEST on page 11-188	0x24	32	RW	0x0	This bits is used to test interrupt from ECC RAM to GIC
MODSTAT on page 11-189	0x28	32	RW	0x0	Counter feature status flag
DERRADDRA on page 11-189	0x2C	32	RO	0x0	This register shows the address of PORTA current double-bit error. RAM size will determine the maximum number of address bits.
SERRADDRA on page 11-190	0x30	32	RO	0x0	This register shows the address of PORTA current single-bit error. RAM size will determine the maximum number of address bits.

Register	Offset	Width	Access	Reset Value	Description
SERRCNTREG on page 11-191	0x3C	32	RW	0x0	Maximum counter value for single-bit error interrupt
ECC_Addrbus on page 11-192	0x40	32	RW	0x0	MSB bit of address is determined by ADR.
ECC_RData0bus on page 11-192	0x44	32	RO	0x0	Data will be read to this register field.
ECC_RData1bus on page 11-193	0x48	32	RO	0x0	Data will be read to this register field.
ECC_RData2bus on page 11-194	0x4C	32	RO	0x0	Data will be read to this register field.
ECC_RData3bus on page 11-194	0x50	32	RO	0x0	Data will be read to this register field.
ECC_WData0bus on page 11-195	0x54	32	WO	0x0	Data from the register will be written to the RAM.
ECC_WData1bus on page 11-196	0x58	32	WO	0x0	Data from the register will be written to the RAM.
ECC_WData2bus on page 11-196	0x5C	32	WO	0x0	Data from the register will be written to the RAM.
ECC_WData3bus on page 11-197	0x60	32	WO	0x0	Data from the register will be written to the RAM.
ECC_RDataecc0bus on page 11-198	0x64	32	RO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Register	Offset	Width	Access	Reset Value	Description
ECC_RDataecc1bus on page 11-198	0x68	32	RO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.
ECC_WDataecc0bus on page 11-199	0x6C	32	WO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.
ECC_WDataecc1bus on page 11-200	0x70	32	WO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.
ECC_dbytectrl on page 11-201	0x74	32	RW	0x0	Max number of implemented byte enabled is DAT/8
ECC_accctrl on page 11-202	0x78	32	RW	0x0	These bits determine which byte of data/ecc to write to RAM.
ECC_startacc on page 11-203	0x7C	32	RW	0x0	These bits determine which byte of data/ecc to write to RAM.
ECC_wdctrl on page 11-204	0x80	32	RW	0x0	Bits to Enable/Disable Watch Dog Timer
SERRLKUPA0 on page 11-204	0x90	32	RW	0x0	Single-bit error address in LOOKUP TABLE for PORTA.

nandw_ecc Summary

Base Address: 0xFF8C2800

Register Address Offset	Bit Fields															
ecc_nandw_ecc_register-Block																
IP_REV_ID 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SIREV RO 0x0															
CTRL 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															INITA 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							CNT_RSTA 0x0	Reserved							ECC_EN 0x0
INITSTAT 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															INITCOM-PLETEA 0x0
ERRINTEN 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTEN 0x0
ERRINTENS 0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTS 0x0
ERRINTENR 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTR 0x0

Register Address Offset	Bit Fields															
INTMODE 0x1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															INTONCMP 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							INTONOVF 0x0	Reserved							INTMODE 0x0
INTSTAT 0x20	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							DERRPENA 0x0	Reserved							SERRPENA 0x0
INTTEST 0x24	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							TDERRA 0x0	Reserved							TSERRA 0x0
MODSTAT 0x28	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															CMPFLGA 0x0
DERRADDRA 0x2C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved									Address 0x0						
SERRADDRA 0x30	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved									Address 0x0						

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SERRCNTREG 0x3C	SERRCNT 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SERRCNT 0x0															
ECC_Addrbus 0x40	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved									ECC_AddrBUS 0x0						
ECC_RData0bus 0x44	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_RData1bus 0x48	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_RData2bus 0x4C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_RData3bus 0x50	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															

Register Address Offset	Bit Fields															
ECC_WData0bus 0x54	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_WData1bus 0x58	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_WData2bus 0x5C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_WData3bus 0x60	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_RDataecc0bus 0x64	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_RDataecc3BUS 0x0							Reserved	ECC_RDataecc2BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_RDataecc1BUS 0x0							Reserved	ECC_RDataecc0BUS 0x0						
ECC_RDataecc1bus 0x68	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_RDataecc7BUS 0x0							Reserved	ECC_RDataecc6BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_RDataecc5BUS 0x0							Reserved	ECC_RDataecc4BUS 0x0						

Register Address Offset	Bit Fields															
ECC_WDataecc0bus 0x6C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_WDataecc3BUS 0x0							Reserved	ECC_WDataecc2BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_WDataecc1BUS 0x0							Reserved	ECC_WDataecc0BUS 0x0						
ECC_WDataecc1bus 0x70	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_WDataecc7BUS 0x0							Reserved	ECC_WDataecc6BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_WDataecc5BUS 0x0							Reserved	ECC_WDataecc4BUS 0x0						
ECC_dbytectrl 0x74	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															DBEN 0x0
ECC_accctrl 0x78	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							RDWR 0x0	Reserved						ECCOVR 0x0	DATAOVR 0x0
ECC_startacc 0x7C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															ENBUSA 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															
ECC_wdctrl 0x80	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															WDEN_RAM 0x0

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SERRLKUPA0 0x90	VALID RW 0x0	Reserved														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved									Address RO 0x0						

IP_REV_ID

Module Instance	Base Address	Register Address
ecc_nandw_ecc_registerBlock	0xFF8C2800	0xFF8C2800

Offset: 0x0

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SIREV RO 0x0															

IP_REV_ID Fields

Bit	Name	Description	Access	Reset
15:0	SIREV	IP Rev # These bits indicate the silicon revision number.	RO	0x0

CTRL

ECC Control Register

Module Instance	Base Address	Register Address
ecc_nandw_ecc_registerBlock	0xFF8C2800	0xFF8C2808

Offset: 0x8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															INITA 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							CNT_RSTA 0x0	Reserved							ECC_EN 0x0

CTRL Fields

Bit	Name	Description	Access	Reset
16	INITA	Enable for the hardware memory initialization PORTA.	RW	0x0
8	CNT_RSTA	Enable to reset internal single-bit error counter A value to zero	RW	0x0
0	ECC_EN	Enable for the ECC detection and correction logic.	RW	0x0

INITSTAT

This bit is used to set the initialize the memory and ecc to a known value

Module Instance	Base Address	Register Address
ecc_nandw_ecc_registerBlock	0xFF8C2800	0xFF8C280C

Offset: 0xC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															INITCOM-PLETEA 0x0

INITSTAT Fields

Bit	Name	Description	Access	Reset
0	INITCOMPLETEA	This bit is used to verify if the hardware memory initialization has completed PORTB.	RW	0x0

ERRINTEN

Error Interrupt enable

Module Instance	Base Address	Register Address
ecc_nandw_ecc_registerBlock	0xFF8C2800	0xFF8C2810

Offset: 0x10

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															SERRINTEN 0x0

ERRINTEN Fields

Bit	Name	Description	Access	Reset
0	SERRINTEN	This bit is used to enable the single bit error interrupt of ECC RAM system	RW	0x0

ERRINTENS

Error Interrupt set

Module Instance	Base Address	Register Address
ecc_nandw_ecc_registerBlock	0xFF8C2800	0xFF8C2814

Offset: 0x14

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															SERRINTS 0x0

ERRINTENS Fields

Bit	Name	Description	Access	Reset
0	SERRINTS	This bit is used to set the single-bit error interrupt bit.	RW	0x0

ERRINTENR

Error Interrupt reset.

Module Instance	Base Address	Register Address
ecc_nandw_ecc_registerBlock	0xFF8C2800	0xFF8C2818

Offset: 0x18

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															SERRINTR 0x0

ERRINTENR Fields

Bit	Name	Description	Access	Reset
0	SERRINTR	This bit is used to reset the single-bit error interrupt bit. Reads reflect SERRINTEN. 1'b0: Writing of zero has no effect. 1'b1: By writing one, this bit will reset SERRINTEN bit to 0. This is performing a bitwise writing of this feature.	RW	0x0

INTMODE

Reads reflect SERRINTEN.

Module Instance	Base Address	Register Address
ecc_nandw_ecc_registerBlock	0xFF8C2800	0xFF8C281C

Offset: 0x1C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															INTONCMP 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							INTONOVF 0x0	Reserved							INTMODE 0x0

INTMODE Fields

Bit	Name	Description	Access	Reset
16	INTONCMP	Enable interrupt on compare.	RW	0x0
8	INTONOVF	Enable interrupt on overflow.	RW	0x0
0	INTMODE	Interrupt mode for single-bit errors.	RW	0x0

INTSTAT

This bit is used to enable interrupt generation on SERR lookup table overflow. When all the entries in the table are valid=1 and this is bit is enabled, serr_req signal will be asserted.

Module Instance	Base Address	Register Address
ecc_nandw_ecc_registerBlock	0xFF8C2800	0xFF8C2820

Offset: 0x20

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							DERRPENA 0x0	Reserved							SERRPENA 0x0

INTSTAT Fields

Bit	Name	Description	Access	Reset
8	DERRPENA	Double-bit error pending for PORTA.	RW	0x0
0	SERRPENA	Single-bit error pending for PORTA.	RW	0x0

INTTEST

This bits is used to test interrupt from ECC RAM to GIC

Module Instance	Base Address	Register Address
ecc_nandw_ecc_registerBlock	0xFF8C2800	0xFF8C2824

Offset: 0x24

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							TDERRA 0x0	Reserved							TSERRA 0x0

INTTEST Fields

Bit	Name	Description	Access	Reset
8	TDERRA	Test PORTA Double-bit error.	RW	0x0
0	TSERRA	Test PORTA Single-bit error.	RW	0x0

MODSTAT

Counter feature status flag

Module Instance	Base Address	Register Address
ecc_nandw_ecc_registerBlock	0xFF8C2800	0xFF8C2828

Offset: 0x28

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															CMPFLGA 0x0

MODSTAT Fields

Bit	Name	Description	Access	Reset
0	CMPFLGA	Port A compare status flag	RW	0x0

DERRADDRA

This register shows the address of PORTA current double-bit error. RAM size will determine the maximum number of address bits.

Module Instance	Base Address	Register Address
ecc_nandw_ecc_registerBlock	0xFF8C2800	0xFF8C282C

Offset: 0x2C

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									Address 0x0						

DERRADDRA Fields

Bit	Name	Description	Access	Reset
6:0	Address	Recent double-bit error address.	RO	0x0

SERRADDRA

This register shows the address of PORTA current single-bit error. RAM size will determine the maximum number of address bits.

Module Instance	Base Address	Register Address
ecc_nandw_ecc_registerBlock	0xFF8C2800	0xFF8C2830

Offset: 0x30

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									Address 0x0						

SERRADDRA Fields

Bit	Name	Description	Access	Reset
6:0	Address	Recent single-bit error address.	RO	0x0

SERRCNTREG

Maximum counter value for single-bit error interrupt

Module Instance	Base Address	Register Address
ecc_nandw_ecc_registerBlock	0xFF8C2800	0xFF8C283C

Offset: 0x3C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SERRCNT 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SERRCNT 0x0															

SERRCNTREG Fields

Bit	Name	Description	Access	Reset
31:0	SERRCNT	Counter value	RW	0x0

ECC_Addrbus

MSB bit of address is determined by ADR.

Module Instance	Base Address	Register Address
ecc_nandw_ecc_registerBlock	0xFF8C2800	0xFF8C2840

Offset: 0x40

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									ECC_AddrBUS 0x0						

ECC_Addrbus Fields

Bit	Name	Description	Access	Reset
6:0	ECC_AddrBUS	Address will be driven to RAM to either read or write the data. Address will be latched by the RAM when the Enbus is asserted.	RW	0x0

ECC_RData0bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_nandw_ecc_registerBlock	0xFF8C2800	0xFF8C2844

Offset: 0x44

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData0bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_RDataBUS	ECC_RDataBUS[31:0].	RO	0x0

ECC_RData1bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_nandw_ecc_registerBlock	0xFF8C2800	0xFF8C2848

Offset: 0x48

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData1bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_RDataBUS	ECC_RDataBUS[63:32].	RO	0x0

ECC_RData2bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_nandw_ecc_registerBlock	0xFF8C2800	0xFF8C284C

Offset: 0x4C

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData2bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_RDataBUS	ECC_RDataBUS[95:64].	RO	0x0

ECC_RData3bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_nandw_ecc_registerBlock	0xFF8C2800	0xFF8C2850

Offset: 0x50

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData3bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_RDataBUS	ECC_RDataBUS[127-96].	RO	0x0

ECC_WData0bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_nandw_ecc_registerBlock	0xFF8C2800	0xFF8C2854

Offset: 0x54

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData0bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_WDataBUS	ECC_WDataBUS[31:0].	WO	0x0

ECC_WData1bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_nandw_ecc_registerBlock	0xFF8C2800	0xFF8C2858

Offset: 0x58

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData1bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_WDataBUS	ECC_WDataBUS[63:32].	WO	0x0

ECC_WData2bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_nandw_ecc_registerBlock	0xFF8C2800	0xFF8C285C

Offset: 0x5C

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData2bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_WDataBUS	ECC_WDataBUS[95-64].	WO	0x0

ECC_WData3bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_nandw_ecc_registerBlock	0xFF8C2800	0xFF8C2860

Offset: 0x60

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData3bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_WDataBUS	ECC_WDataBUS[127-96].	WO	0x0

ECC_RDataecc0bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_nandw_ecc_registerBlock	0xFF8C2800	0xFF8C2864

Offset: 0x64

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	ECC_RDataecc3BUS 0x0							Reserved	ECC_RDataecc2BUS 0x0						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	ECC_RDataecc1BUS 0x0							Reserved	ECC_RDataecc0BUS 0x0						

ECC_RDataecc0bus Fields

Bit	Name	Description	Access	Reset
30:24	ECC_RDataecc3BUS	Eccdata will be read to this register field.	RO	0x0
22:16	ECC_RDataecc2BUS	Eccdata will be read to this register field.	RO	0x0
14:8	ECC_RDataecc1BUS	Eccdata will be read to this register field.	RO	0x0
6:0	ECC_RDataecc0BUS	Eccdata will be read to this register field.	RO	0x0

ECC_RDataecc1bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_nandw_ecc_registerBlock	0xFF8C2800	0xFF8C2868

Offset: 0x68

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	ECC_RDataecc7BUS 0x0							Reserved	ECC_RDataecc6BUS 0x0						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	ECC_RDataecc5BUS 0x0							Reserved	ECC_RDataecc4BUS 0x0						

ECC_RDataecc1bus Fields

Bit	Name	Description	Access	Reset
30:24	ECC_RDataecc7BUS	Eccdata will be read to this register field.	RO	0x0
22:16	ECC_RDataecc6BUS	Eccdata will be read to this register field.	RO	0x0
14:8	ECC_RDataecc5BUS	Eccdata will be read to this register field.	RO	0x0
6:0	ECC_RDataecc4BUS	Eccdata will be read to this register field.	RO	0x0

ECC_WDataecc0bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_nandw_ecc_registerBlock	0xFF8C2800	0xFF8C286C

Offset: 0x6C

Access: WO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	ECC_WDataecc3BUS 0x0							Reserved	ECC_WDataecc2BUS 0x0						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	ECC_WDataecc1BUS 0x0							Reserved	ECC_WDataecc0BUS 0x0						

ECC_WDataecc0bus Fields

Bit	Name	Description	Access	Reset
30:24	ECC_WDataecc3BUS	Eccdata from the register will be written to the RAM.	WO	0x0
22:16	ECC_WDataecc2BUS	Eccdata from the register will be written to the RAM.	WO	0x0
14:8	ECC_WDataecc1BUS	Eccdata from the register will be written to the RAM.	WO	0x0
6:0	ECC_WDataecc0BUS	Eccdata from the register will be written to the RAM.	WO	0x0

ECC_WDataecc1bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_nandw_ecc_registerBlock	0xFF8C2800	0xFF8C2870

Offset: 0x70

Access: WO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	ECC_WDataecc7BUS 0x0							Reserved	ECC_WDataecc6BUS 0x0						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	ECC_WDataecc5BUS 0x0							Reserved	ECC_WDataecc4BUS 0x0						

ECC_WDataecc1bus Fields

Bit	Name	Description	Access	Reset
30:24	ECC_WDataecc7BUS	Eccdata from the register will be written to the RAM.	WO	0x0
22:16	ECC_WDataecc6BUS	Eccdata from the register will be written to the RAM.	WO	0x0
14:8	ECC_WDataecc5BUS	Eccdata from the register will be written to the RAM.	WO	0x0
6:0	ECC_WDataecc4BUS	Eccdata from the register will be written to the RAM.	WO	0x0

ECC_dbytectrl

ECC Data Byte Control. The maximum number of implemented byte enables equals the data width of the RAM divided by 8.

Module Instance	Base Address	Register Address
ecc_nandw_ecc_registerBlock	0xFF8C2800	0xFF8C2874

Offset: 0x74

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															DBEN 0x0

ECC_dbytectrl Fields

Bit	Name	Description	Access	Reset
0	DBEN	Byte or word enable for access.	RW	0x0

ECC_accctrl

These bits determine which byte of data/ecc to write to RAM.

Module Instance	Base Address	Register Address
ecc_nandw_ecc_registerBlock	0xFF8C2800	0xFF8C2878

Offset: 0x78

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							RDWR 0x0	Reserved						ECCOVR 0x0	DATAOVR 0x0

ECC_accctrl Fields

Bit	Name	Description	Access	Reset
8	RDWR	Control for read/write.	RW	0x0
1	ECCOVR	ECC Data Override.	RW	0x0
0	DATAOVR	RAM Data Override. Override the ECC_dataBUS register with RAM data in read mode set by ECC_RW. 1'b0: Data override disabled. 1'b1: Data override enabled.	RW	0x0

ECC_startacc

These bits determine which byte of data/ecc to write to RAM.

Module Instance	Base Address	Register Address
ecc_nandw_ecc_registerBlock	0xFF8C2800	0xFF8C287C

Offset: 0x7C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															ENBUSA 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															

ECC_startacc Fields

Bit	Name	Description	Access	Reset
16	ENBUSA	Start RAM access for PORTA.	RW	0x0

ECC_wdctrl

Bits to Enable/Disable Watch Dog Timer

Module Instance	Base Address	Register Address
ecc_nandw_ecc_registerBlock	0xFF8C2800	0xFF8C2880

Offset: 0x80

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															WDEN_RAM 0x0

ECC_wdctrl Fields

Bit	Name	Description	Access	Reset
0	WDEN_RAM	Enable watchdog timeout for OCP register access to IP RAM.	RW	0x0

SERRLKUPA0

Single-bit error address in LOOKUP TABLE for PORTA.

Module Instance	Base Address	Register Address
ecc_nandw_ecc_registerBlock	0xFF8C2800	0xFF8C2890

Offset: 0x90

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VALID RW 0x0	Reserved														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									Address RO 0x0						

SERRLKUPA0 Fields

Bit	Name	Description	Access	Reset
31	VALID	Valid flag bit. Valid bit indicates if the address in this register is current or stale.	RW	0x0
6:0	Address	Recent Single-bit error address. This register shows the address of the each single-bit error. RAM size will determine the maximum number of address bits. If ram size is 32 Kbytes, bit 30-16 will be reserved and read as zero.	RO	0x0

nand_config Address Map

Module Instance	Base Address	End Address
i_nand_config	0xFFB80000	0xFFB802FF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
transfer_spare_reg on page 13-142	0x10	32	RW	0x0	Default data transfer mode. (Ignored during Spare only mode)
load_wait_cnt on page 13-142	0x20	32	RW	0x1F4	Wait count value for Load operation
program_wait_cnt on page 13-144	0x30	32	RW	0x1F40	Wait count value for Program operation
erase_wait_cnt on page 13-145	0x40	32	RW	0x1F40	Wait count value for Erase operation
int_mon_cyccnt on page 13-146	0x50	32	RW	0x1F4	Interrupt monitor cycle count value

Register	Offset	Width	Access	Reset Value	Description
rb_pin_enabled on page 13-147	0x60	32	RW	0x1	Interrupt or polling mode. Ready/Busy pin is enabled from device.
multiplane_operation on page 13-149	0x70	32	RW	0x0	Multiplane transfer mode. Pipelined read, copyback, erase and program commands are transfered in multiplane mode
multiplane_read_enable on page 13-150	0x80	32	RW	0x0	Device supports multiplane read command sequence
copyback_disable on page 13-151	0x90	32	RW	0x0	Device does not support copyback command sequence
cache_write_enable on page 13-152	0xA0	32	RW	0x0	Device supports cache write command sequence
cache_read_enable on page 13-153	0xB0	32	RW	0x0	Device supports cache read command sequence
prefetch_mode on page 13-153	0xC0	32	RW	0x1	Enables read data prefetching to faster performance
chip_enable_dont_care on page 13-155	0xD0	32	RW	0x0	Device can work in the chip enable dont care mode
ecc_enable on page 13-155	0xE0	32	RW	0x1	Enable controller ECC check bit generation and correction
global_int_enable on page 13-156	0xF0	32	RW	0x0	Global Interrupt enable and Error/ Timeout disable.
twhr2_and_we_2_re on page 13-157	0x100	32	RW	0x1432	
tcwaw_and_addr_2_data on page 13-158	0x110	32	RW	0x1432	

Register	Offset	Width	Access	Reset Value	Description
re_2_we on page 13-159	0x120	32	RW	0x32	Timing parameter between re high to we low (Trhw)
acc_clks on page 13-160	0x130	32	RW	0x0	Timing parameter from read enable going low to capture read data
number_of_planes on page 13-161	0x140	32	RW	0x0	Number of planes in the device
pages_per_block on page 13-162	0x150	32	RW	0x0	Number of pages in a block
device_width on page 13-163	0x160	32	RW	0x3	I/O width of attached devices
device_main_area_size on page 13-164	0x170	32	RW	0x0	Page main area size of device in bytes
device_spare_area_size on page 13-165	0x180	32	RW	0x0	Page spare area size of device in bytes
two_row_addr_cycles on page 13-166	0x190	32	RW	0x0	Attached device has only 2 ROW address cycles
multiplane_addr_restrict on page 13-167	0x1A0	32	RW	0x0	Address restriction for multiplane commands
ecc_correction on page 13-168	0x1B0	32	RW	0x8	Correction capability required and the Erase threshold value.
read_mode on page 13-171	0x1C0	32	RW	0x0	The type of read sequence that the controller will follow for pipe read commands.
write_mode on page 13-173	0x1D0	32	RW	0x0	The type of write sequence that the controller will follow for pipe write commands.
copyback_mode on page 13-174	0x1E0	32	RW	0x0	The type of copyback sequence that the controller will follow.

Register	Offset	Width	Access	Reset Value	Description
rdwr_en_lo_cnt on page 13-176	0x1F0	32	RW	0x12	Read/Write Enable low pulse width
rdwr_en_hi_cnt on page 13-177	0x200	32	RW	0xC	Read/Write Enable high pulse width
max_rd_delay on page 13-178	0x210	32	RW	0x0	Max round trip read data delay for data capture
cs_setup_cnt on page 13-179	0x220	32	RW	0xA003	Chip select setup/tWB time
spare_area_skip_bytes on page 13-180	0x230	32	RW	0x0	Spare area skip bytes
spare_area_marker on page 13-181	0x240	32	RW	0xFFFF	Spare area marker value
devices_connected on page 13-182	0x250	32	RW	0x0	Number of Devices connected on one bank
die_mask on page 13-183	0x260	32	RW	0x0	Indicates the die differentiator in case of NAND devices with stacked dies.
first_block_of_next_plane on page 13-184	0x270	32	RW	0x1	The starting block address of the next plane in a multi plane device.
write_protect on page 13-185	0x280	32	RW	0x1	This register is used to control the assertion/de-assertion of the WP# pin to the device.
re_2_re on page 13-186	0x290	32	RW	0x32	Timing parameter between re high to re low (Trhz) for the next bank
por_reset_count on page 13-187	0x2A0	32	RW	0x13B	The number of cycles the controller waits after POR to issue the first RESET command to the device.

Register	Offset	Width	Access	Reset Value	Description
watchdog_reset_count on page 13-188	0x2B0	32	RW	0x5B9A	The number of cycles the controller waits before flagging a watchdog timeout interrupt.
device_reset on page 13-189	0x0	32	RW	0x0	Device reset. Controller sends a RESET command to device. Controller resets bit after sending command to device

nand_config Summary

Base Address: 0xFFB80000

Register Address Offset	Bit Fields															
i_nand_config																
transfer_spare_reg 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															flag RW 0x0
load_wait_cnt 0x20	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	value RW 0x1F4															
program_wait_cnt 0x30	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	value RW 0x1F40															

Register Address Offset	Bit Fields															
erase_wait_cnt 0x40	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	value RW 0x1F40															
int_mon_cyc cnt 0x50	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	value RW 0x1F4															
rb_pin_enab led 0x60	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved												bank 3 RW 0x0	bank 2 RW 0x0	bank 1 RW 0x0	bank0 RW 0x1
multiplane_ operation 0x70	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															flag RW 0x0
multiplane_ read_enable 0x80	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															flag RW 0x0
copyback_di sable 0x90	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															flag RW 0x0
cache_write _enable 0xA0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															flag RW 0x0

Register Address Offset	Bit Fields															
cache_read_enable 0xB0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															flag RW 0x0
prefetch_mode 0xC0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	prefetch_burst_length RW 0x0												Reserved			prefetch_en RW 0x1
chip_enable_dont_care 0xD0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															flag RW 0x0
ecc_enable 0xE0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															flag RW 0x1
global_int_enable 0xF0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							error_rpt_disable RW 0x0	Reserved			timeout_disable RW 0x0	Reserved			flag RW 0x0
twhr2_and_we_2_re 0x100	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		twhr2 RW 0x14						Reserved		we_2_re RW 0x32					

Register Address Offset	Bit Fields															
tcwaw_and_addr_2_data 0x110	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		tcwaw RW 0x14						Reserved	addr_2_data RW 0x32						
re_2_we 0x120	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved										value RW 0x32					
acc_clks 0x130	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved												value RW 0x0			
number_of_planes 0x140	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													value RW 0x0		
pages_per_block 0x150	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	value RW 0x0															
device_width 0x160	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														value RW 0x3	
device_main_area_size 0x170	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	value RW 0x0															

Register Address Offset	Bit Fields															
device_spare_area_size 0x180	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	value RW 0x0															
two_row_addr_cycles 0x190	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved											four RW 0x0	Reserved			flag RW 0x0
multiplane_addr_restrict 0x1A0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															flag RW 0x0
ecc_correction 0x1B0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	erase_threshold RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								value RW 0x8							
read_mode 0x1C0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved												value RW 0x0			
write_mode 0x1D0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved												value RW 0x0			
copyback_mode 0x1E0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved												value RW 0x0			

Register Address Offset	Bit Fields															
rdwr_en_lo_cnt 0x1F0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved											value RW 0x12				
rdwr_en_hi_cnt 0x200	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved											value RW 0xC				
max_rd_delay 0x210	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved												value RW 0x0			
cs_setup_cnt 0x220	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved														twb RW 0xA	
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	twb RW 0xA				Reserved							value RW 0x3				
spare_area_skip_bytes 0x230	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved										value RW 0x0					
spare_area_marker 0x240	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	value RW 0xFFFF															
devices_connected 0x250	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													value RW 0x0		

Register Address Offset	Bit Fields															
die_mask 0x260	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	value RW 0x0															
first_block_of_next_plane 0x270	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	value RW 0x1															
write_protect 0x280	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															flag RW 0x1
re_2_re 0x290	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved										value RW 0x32					
por_reset_count 0x2A0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	value RW 0x13B															
watchdog_reset_count 0x2B0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	value RW 0x5B9A															
device_reset 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved												bank 3 RW 0x0	bank 2 RW 0x0	bank 1 RW 0x0	bank0 RW 0x0

transfer_spare_reg

Default data transfer mode. (Ignored during Spare only mode)

Module Instance	Base Address	Register Address
i_nand_config	0xFFB80000	0xFFB80010

Offset: 0x10

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															flag RW 0x0

transfer_spare_reg Fields

Bit	Name	Description	Access	Reset
0	flag	On all read or write commands through Map 01, if this bit is set, data in spare area of memory will be transfered to host along with main area of data. The main area will be transfered followed by spare area.[list][*]1 - MAIN+SPARE [*]0 - MAIN[/list]	RW	0x0

load_wait_cnt

Wait count value for Load operation

Module Instance	Base Address	Register Address
i_nand_config	0xFFB80000	0xFFB80020

Offset: 0x20

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
value RW 0x1F4															

load_wait_cnt Fields

Bit	Name	Description	Access	Reset
15:0	value	Number of clock cycles after issue of load operation before Cadence NAND Flash Controller polls for status. This values is of relevance for status polling mode of operation and has been provided to minimize redundant polling after issuing a command. After a load command, the first polling will happen after this many number of cycles have elapsed and then on polling will happen every intmon_cyc_cnt cycles. The default values is equal to the default value of intmon_cyc_cnt.	RW	0x1F4

program_wait_cnt

Wait count value for Program operation

Module Instance	Base Address	Register Address
i_nand_config	0xFFB80000	0xFFB80030

Offset: 0x30

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
value RW 0x1F40															

program_wait_cnt Fields

Bit	Name	Description	Access	Reset
15:0	value	Number of clock cycles after issue of program operation before Cadence NAND Flash Controller polls for status. This values is of relevance for status polling mode of operation and has been provided to minimize redundant polling after issuing a command. After a program command, the first polling will happen after this many number of cycles have elapsed and then on polling will happen every intmon_cyc_cnt cycles. The default values is equal to the default value of intmon_cyc_cnt. The controller internally multiplies the value programmed into this register by 16 to provide a wider range for polling.	RW	0x1F40

erase_wait_cnt

Wait count value for Erase operation

Module Instance	Base Address	Register Address
i_nand_config	0xFFB80000	0xFFB80040

Offset: 0x40

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
value RW 0x1F40															

erase_wait_cnt Fields

Bit	Name	Description	Access	Reset
15:0	value	Number of clock cycles after issue of erase operation before Cadence NAND Flash Controller polls for status. This values is of relevance for status polling mode of operation and has been provided to minimize redundant polling after issuing a command. After a erase command, the first polling will happen after this many number of cycles have elapsed and then on polling will happen every intmon_cyc_cnt cycles. The default values is equal to the default value of intmon_cyc_cnt. The controller internally multiplies the value programmed into this register by 16 to provide a wider range for polling.	RW	0x1F40

int_mon_cyccnt

Interrupt monitor cycle count value

Module Instance	Base Address	Register Address
i_nand_config	0xFFB80000	0xFFB80050

Offset: 0x50

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
value RW 0x1F4															

int_mon_cyccnt Fields

Bit	Name	Description	Access	Reset
15:0	value	In polling mode, sets the number of cycles Cadence Flash Controller must wait before checking the status register. This register is only used when R/B pins are not available to Cadence NAND Flash Controller.	RW	0x1F4

rb_pin_enabled

Interrupt or polling mode. Ready/Busy pin is enabled from device.

Module Instance	Base Address	Register Address
i_nand_config	0xFFB80000	0xFFB80060

Offset: 0x60

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												bank3 RW 0x0	bank2 RW 0x0	bank1 RW 0x0	bank0 RW 0x1

rb_pin_enabled Fields

Bit	Name	Description	Access	Reset
3	bank3	Sets Cadence Flash Controller in interrupt pin or polling mode [list][*]1 - R/B pin enabled for bank 3. Interrupt pin mode. [*]0 - R/B pin disabled for bank 3. Polling mode.[/list]	RW	0x0
2	bank2	Sets Cadence Flash Controller in interrupt pin or polling mode [list][*]1 - R/B pin enabled for bank 2. Interrupt pin mode. [*]0 - R/B pin disabled for bank 2. Polling mode.[/list]	RW	0x0
1	bank1	Sets Cadence Flash Controller in interrupt pin or polling mode [list][*]1 - R/B pin enabled for bank 1. Interrupt pin mode. [*]0 - R/B pin disabled for bank 1. Polling mode.[/list]	RW	0x0

Bit	Name	Description	Access	Reset
0	bank0	Sets Cadence Flash Controller in interrupt pin or polling mode [list][*]1 - R/B pin enabled for bank 0. Interrupt pin mode. [*]0 - R/B pin disabled for bank 0. Polling mode.[/list]	RW	0x1

multiplane_operation

Multiplane transfer mode. Pipelined read, copyback, erase and program commands are transfered in multiplane mode

Module Instance	Base Address	Register Address
i_nand_config	0xFFB80000	0xFFB80070

Offset: 0x70

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															flag RW 0x0

multiplane_operation Fields

Bit	Name	Description	Access	Reset
0	flag	[list][*]1 - Multiplane operation enabled [*]0 - Multiplane operation disabled[/list]	RW	0x0

multiplane_read_enable

Device supports multiplane read command sequence

Module Instance	Base Address	Register Address
i_nand_config	0xFFB80000	0xFFB80080

Offset: 0x80

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															flag RW 0x0

multiplane_read_enable Fields

Bit	Name	Description	Access	Reset
0	flag	Certain devices support dedicated multiplane read command sequences to read data in the same fashion as is written with multiplane program commands. This bit set should be set for the above devices. When not set, pipeline reads in multiplane mode will still happen in the order of multiplane writes, though normal read command sequences will be issued to the device. [list][*]1 - Device supports multiplane read sequence [*]0 - Device does not support multiplane read sequence[/list]	RW	0x0

copyback_disable

Device does not support copyback command sequence

Module Instance	Base Address	Register Address
i_nand_config	0xFFB80000	0xFFB80090

Offset: 0x90

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															flag RW 0x0

copyback_disable Fields

Bit	Name	Description	Access	Reset
0	flag	[list][*]1 - Copyback disabled [*]0 - Copyback enabled[/list]	RW	0x0

cache_write_enable

Device supports cache write command sequence

Module Instance	Base Address	Register Address
i_nand_config	0xFFB80000	0xFFB800A0

Offset: 0xA0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															flag RW 0x0

cache_write_enable Fields

Bit	Name	Description	Access	Reset
0	flag	[list][*]1 - Cache write supported [*]0 - Cache write not supported[/list]	RW	0x0

cache_read_enable

Device supports cache read command sequence

Module Instance	Base Address	Register Address
i_nand_config	0xFFB80000	0xFFB800B0

Offset: 0xB0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															flag RW 0x0

cache_read_enable Fields

Bit	Name	Description	Access	Reset
0	flag	[list][*]1 - Cache read supported [*]0 - Cache read not supported[/list]	RW	0x0

prefetch_mode

Enables read data prefetching to faster performance

Module Instance	Base Address	Register Address
i_nand_config	0xFFB80000	0xFFB800C0

Offset: 0xC0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
prefetch_burst_length RW 0x0												Reserved			prefetch_en RW 0x1

prefetch_mode Fields

Bit	Name	Description	Access	Reset
15:4	prefetch_burst_length	If prefetch_en is set and prefetch_burst_length is set to ZERO, the controller will start prefetching data only after the receiving the first Map01 read command for the page. If prefetch_en is set and prefetch_burst_length is set to a non-ZERO, valid value, the controller will start prefetching data corresponding to this value even before the first Map01 for the current page has been received. The value written here should be in bytes.	RW	0x0
0	prefetch_en	Enable prefetch of Data	RW	0x1

chip_enable_dont_care

Device can work in the chip enable dont care mode

Module Instance	Base Address	Register Address
i_nand_config	0xFFB80000	0xFFB800D0

Offset: 0xD0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															flag RW 0x0

chip_enable_dont_care Fields

Bit	Name	Description	Access	Reset
0	flag	Controller can interleave commands between banks when this feature is enabled. [list][*]1 - Device in dont care mode [*]0 - Device cares for chip enable[/list]	RW	0x0

ecc_enable

Enable controller ECC check bit generation and correction

Module Instance	Base Address	Register Address
i_nand_config	0xFFB80000	0xFFB800E0

Offset: 0xE0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															flag RW 0x1

ecc_enable Fields

Bit	Name	Description	Access	Reset
0	flag	Enables or disables controller ECC capabilities. When enabled, controller calculates ECC check-bits and writes them onto device on program operation. On page reads, check-bits are recomputed and errors reported, if any, after comparing with stored check-bits. When disabled, controller does not compute check-bits. [list] [*]1 - ECC Enabled [*]0 - ECC disabled[/list]	RW	0x1

global_int_enable

Global Interrupt enable and Error/Timeout disable.

Module Instance	Base Address	Register Address
i_nand_config	0xFFB80000	0xFFB800F0

Offset: 0xF0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							error_rpt_disable RW 0x0	Reserved			timeout_disable RW 0x0	Reserved			flag RW 0x0

global_int_enable Fields

Bit	Name	Description	Access	Reset
8	error_rpt_disable	Command and ECC uncorrectable failures will not be reported when this bit is set	RW	0x0
4	timeout_disable	Watchdog timer logic will be de-activated when this bit is set.	RW	0x0
0	flag	Host will receive an interrupt only when this bit is set.	RW	0x0

twhr2_and_we_2_re

Module Instance	Base Address	Register Address
i_nand_config	0xFFB80000	0xFFB80100

Offset: 0x100

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		twhr2 RW 0x14						Reserved		we_2_re RW 0x32					

twhr2_and_we_2_re Fields

Bit	Name	Description	Access	Reset
13:8	twhr2	Signifies the number of controller clocks that should be introduced between the last command of a random data output command to the start of the data transfer.	RW	0x14
5:0	we_2_re	Signifies the number of bus interface clk_x clocks that should be introduced between write enable going high to read enable going low. The number of clocks is the function of device parameter Twhr and controller clock frequency.	RW	0x32

tcwaw_and_addr_2_data

Module Instance	Base Address	Register Address
i_nand_config	0xFFB80000	0xFFB80110

Offset: 0x110

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		tcwaw RW 0x14						Reserved	addr_2_data RW 0x32						

tcwaw_and_addr_2_data Fields

Bit	Name	Description	Access	Reset
13:8	tcwaw	Signifies the number of controller clocks that should be introduced between the command cycle of a random data input command to the address cycle of the random data input command.	RW	0x14
6:0	addr_2_data	Signifies the number of bus interface clk_x clocks that should be introduced between an address to a data input cycle. The number of clocks is the function of device parameter Tadl and controller clock frequency.	RW	0x32

re_2_we

Timing parameter between re high to we low (Trhw)

Module Instance	Base Address	Register Address
i_nand_config	0xFFB80000	0xFFB80120

Offset: 0x120

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										value RW 0x32					

re_2_we Fields

Bit	Name	Description	Access	Reset
5:0	value	Signifies the number of bus interface clk_x clocks that should be introduced between read enable going high to write enable going low. The number of clocks is the function of device parameter Trhw and controller clock frequency.	RW	0x32

acc_clks

Timing parameter from read enable going low to capture read data

Module Instance	Base Address	Register Address
i_nand_config	0xFFB80000	0xFFB80130

Offset: 0x130

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												value RW 0x0			

acc_clks Fields

Bit	Name	Description	Access	Reset
3:0	value	Signifies the number of bus interface clk_x clock cycles, controller should wait from read enable going low to sending out a strobe of clk_x for capturing of incoming data.	RW	0x0

number_of_planes

Number of planes in the device

Module Instance	Base Address	Register Address
i_nand_config	0xFFB80000	0xFFB80140

Offset: 0x140

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													value RW 0x0		

number_of_planes Fields

Bit	Name	Description	Access	Reset
2:0	value	Controller will read Electronic Signature of devices and populate this field as the number of planes information is present in the signature. For 512B device, this information needs to be programmed by software. Software could also choose to override the populated value. The values in the fields should be as follows[list] [*]3'h0 - Monoplane device [*]3'h1 - Two plane device [*]3'h3 - 4 plane device [*]3'h7 - 8 plane device [*]All other values - Reserved[/list]	RW	0x0

pages_per_block

Number of pages in a block

Module Instance	Base Address	Register Address
i_nand_config	0xFFB80000	0xFFB80150

Offset: 0x150

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
value RW 0x0															

pages_per_block Fields

Bit	Name	Description	Access	Reset
15:0	value	Controller will read Electronic Signature of devices and populate this field. For 512B devices, bootstrap_512B_device will determine the value of this field to be of 32. Software could also choose to override the populated value.	RW	0x0

device_width

I/O width of attached devices

Module Instance	Base Address	Register Address
i_nand_config	0xFFB80000	0xFFB80160

Offset: 0x160

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														value RW 0x3	

device_width Fields

Bit	Name	Description	Access	Reset
1:0	value	Controller will read Electronic Signature of devices and populate this field. For 512B devices, bootstrap_x16_device will determine the value of this field. Software could also choose to override the populated value. The values in this field should be as follows[list][*]2'h00 - 8bit device[*] 2'h01 - 16bit device[*]All other values - Reserved[/list]	RW	0x3

device_main_area_size

Page main area size of device in bytes

Module Instance	Base Address	Register Address
i_nand_config	0xFFB80000	0xFFB80170

Offset: 0x170

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
value RW 0x0															

device_main_area_size Fields

Bit	Name	Description	Access	Reset
15:0	value	Controller will read Electronic Signature of devices and populate this field. For 512B devices, bootstrap_512B_device will determine the value of this field to be 512. Software could also choose to override the populated value.	RW	0x0

device_spare_area_size

Page spare area size of device in bytes

Module Instance	Base Address	Register Address
i_nand_config	0xFFB80000	0xFFB80180

Offset: 0x180

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
value RW 0x0															

device_spare_area_size Fields

Bit	Name	Description	Access	Reset
15:0	value	Controller will read Electronic Signature of devices and populate this field. For 512B devices, bootstrap_512B_device will determine the value of this field to be 16. Software could also choose to override the populated value.	RW	0x0

two_row_addr_cycles

Attached device has only 2 ROW address cycles

Module Instance	Base Address	Register Address
i_nand_config	0xFFB80000	0xFFB80190

Offset: 0x190

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved											four RW 0x0	Reserved			flag RW 0x0

two_row_addr_cycles Fields

Bit	Name	Description	Access	Reset
4	four	This flag must be set for devices which allow for 4 ROW address cycles instead of the usual 3.	RW	0x0
0	flag	This flag must be set for devices which allow for 2 ROW address cycles instead of the usual 3. Alternatively, bootstrap_two_row_addr_cycles when asserted will set this flag.	RW	0x0

multiplane_addr_restrict

Address restriction for multiplane commands

Module Instance	Base Address	Register Address
i_nand_config	0xFFB80000	0xFFB801A0

Offset: 0x1A0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															flag RW 0x0

multiplane_addr_restrict Fields

Bit	Name	Description	Access	Reset
0	flag	This flag must be set for devices which require that during multiplane operations all but the address for the last plane should have their address cycles tied low. The last plane address cycles has proper values. This ensures multiplane address restrictions in the device.	RW	0x0

ecc_correction

Correction capability required and the Erase threshold value.

Module Instance	Base Address	Register Address
i_nand_config	0xFFB80000	0xFFB801B0

Offset: 0x1B0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
erase_threshold RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								value RW 0x8							

ecc_correction Fields

Bit	Name	Description	Access	Reset
31:16	erase_threshold	This value informs the ECC logic of the number of 0's to count in a page before considering it as Erased. If the number of 0's in the page being read is less than the value in this register, an erased page is inferred and no un-correctable error will be flagged for that page. If ECC is disabled, the erased_page interrupt shall be set as explained above. If ECC is enabled, in addition to the above condition, only when the ECC logic detects an un-correctable error for that page will the erased_ page interrupt be flagged. If the ECC logic detects a no-error or correct-able error page, this erased page interrupt will not be set. A value of ZERO in this register will disabled checking for erased pages. Erased page detection logic will be activated only in MAIN or MAIN+SPARE or META-DATA(if available) modes of operation.	RW	0x0

Bit	Name	Description	Access	Reset
7:0	value	The required correction capability. A smaller correction capability will lead to lesser number of ECC check-bits being written per ECC sector. The supported ECC correction levels are - [list] [*] 16,8,4 over 512 bytes. [*] 24 over 1024 bytes. [*] All other values will cause the correction value in the controller to fall back to the previously selected value. [/list]	RW	0x8

read_mode

The type of read sequence that the controller will follow for pipe read commands.

Module Instance	Base Address	Register Address
i_nand_config	0xFFB80000	0xFFB801C0

Offset: 0x1C0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												value RW 0x0			

read_mode Fields

Bit	Name	Description	Access	Reset
3:0	value	The values in the field should be as follows: 4'h0 - This value informs the controller that the pipe read sequence to follow is of a normal read. For 512 byte page devices, Normal read sequence is, C00, Address, Data, ... For devices with page size greater that 512 bytes, the sequence is, C00, Address, C30, Data..... 4'h1 - This value informs the controller that the pipe read sequence to follow is of a Cache Read with the following sequence, C00, Address, C30, C31, Data, C31, Data, ..., C3F, Data. 4'h2 - This value informs the controller that the pipe read sequence to follow is of a Cache Read with the following sequence, C00, Address, C31, Data, Data, ..., C34. 4'h3 - This value informs the controller that the pipe read sequence to follow is of a 'N' Plane Read with the following sequence, C00, Address, C00, Address, C30, Data, C06, Address, CE0, Data..... 4'h4 - This value informs the controller that the pipe read sequence to follow is of a 'N' Plane Read with the following sequence, C60, Address, C60, Address, C30, C00, Address, C05, Address, CE0, Data, C00, Address, C05, Address, CE0, Data..... 4'h5 - This value informs the controller that the pipe read sequence to follow is of a 'N' Plane Cache Read with the following sequence, C60, Address, C60, Address, C30, C31, C00, Address, C05, Address, CE0, Data, C00, Address, C05, Address, CE0, Data, ..., C3F, C00, Address, C05, Address, CE0, Data, C00, Address, C05, Address, CE0, Data 4'h6 - This value informs the controller that the pipe read sequence to follow is of a 'N' Plane Read with the following sequence, C00, Address, C32, .., C00,	RW	0x0

Bit	Name	Description	Access	Reset
		Address, C30, C06, Address, CE0, Data, C06, Address, CE0, Data,.... 4'h7 - This value informs the controller that the pipe read sequence to follow is of a 'N' Plane Cache Read with the following sequence, C00, Address, C32,..., C00, Address, C30, C31,C06, Address, CE0, Data, C31, C06, Address, CE0, Data, C3F, C06, Address, CE0, Data.... 4'h8 - This value informs the controller that the pipe read sequence to follow is of a 'N' Plane Cache Read with the following sequence, C60, Address, C60, Address, C33, C31, C00, Address, C05, Address, CE0, Data, C00, Address, C05, Address, CE0, Data, ..., C3F, C00, Address, C05, Address, CE0, Data, C00, Address, C05, Address, CE0, Data 4'h9 - 4'h15 - Reserved. ..... indicates that the previous sequence is repeated till the last page.		

write_mode

```
        The type of write sequence that the controller will follow
for pipe write commands.
```

Module Instance	Base Address	Register Address
i_nand_config	0xFFB80000	0xFFB801D0

Offset: 0x1D0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												value RW 0x0			

write_mode Fields

Bit	Name	Description	Access	Reset
3:0	value	The values in the field should be as follows: 4'h0 - This value informs the controller that the pipe write sequence to follow is of a normal write with the following sequence, C80, Address, Data, C10..... 4'h1 - This value informs the controller that the pipe write sequence to follow is of a Cache Program with the following sequence, C80, Address, Data, C15, ..., C80, Address, Data, C10. 4'h2 - This value informs the controller that the pipe write sequence to follow is of a Two/Four Plane Program with the following sequence, C80, Address, Data, C11, C81, Address, Data, C10..... 4'h3 - This value informs the controller that the pipe write sequence to follow is of a 'N' Plane Program with the following sequence, C80, Address, Data, C11, C80, Address, Data, C10..... 4'h4 - This value informs the controller that the pipe write sequence to follow is of a 'N' Plane Cache Program with the following sequence, C80, Address, Data, C11, C80, Address, Data, C15.....C80, Address, Data, C11, C80, Address, Data, C10. 4'h5 - This value informs the controller that the pipe write sequence to follow is of a 'N' Plane Cache Program with the following sequence, C80, Address, Data, C11, C81, Address, Data, C15.....C80, Address, Data, C11, C81, Address, Data, C10. 4'h6 - 4'h15 - Reserved. ..... indicates that the previous sequence is repeated till the last page.	RW	0x0

copyback_mode

The type of copyback sequence that the controller will follow.

Module Instance	Base Address	Register Address
i_nand_config	0xFFB80000	0xFFB801E0

Offset: 0x1E0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												value RW 0x0			

copyback_mode Fields

Bit	Name	Description	Access	Reset
3:0	value	The values in the field should be as follows 4'h0 - This value informs the controller that the copyback sequence to follow is, C00, Address, C35, C85, Address, C10 4'h1 - This value informs the controller that the copyback sequence to follow is, C00, Address, C30, C8C, Address, C10 4'h2 - This value informs the controller that the copyback sequence to follow is, C00, Address, C8A, Address, C10 4'h3 - This value informs the controller that the copyback sequence to follow is of a four plane copyback sequence, C00, Address, C03, Address, C03, Address, C03, Address, C8A, Address, C11, C8A, Address, C11, C8A, Address, C11, C8A, Address, C10. 4'h4 - This value informs the controller that the copyback sequence to follow is of a two plane copyback sequence, C00, Address, C35, C00, Address, C35, C85, Address, C11, C81, Address, C10. 4'h5 - This value informs the controller that the copyback sequence to follow is of a two plane copyback sequence, C60, Address, C60, Address, C35, C85, Address, C11, C81, Address, C10. 4'h6 - This value informs the controller that the copyback sequence to follow is of a two plane copyback sequence, C00, Address, C00, Address, C35, C85, Address, C11, C80, Address, C10.	RW	0x0

Bit	Name	Description	Access	Reset
		4'h7 - This value informs the controller that the copyback sequence to follow is of a two plane copyback sequence, C60, Address, C60, Address, C30, C8C, Address, C11, C8C, Address, C10. 4'h8 - 4'h15 - Reserved.		

rdwr_en_lo_cnt

Read/Write Enable low pulse width

Module Instance	Base Address	Register Address
i_nand_config	0xFFB80000	0xFFB801F0

Offset: 0x1F0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved											value RW 0x12				

rdwr_en_lo_cnt Fields

Bit	Name	Description	Access	Reset
4:0	value	Number of clk_x cycles that read or write enable will kept low to meet the min Trp/Twp parameter of the device. The value in this register plus rdwr_en_hi_cnt register value should meet the min cycle time of the device connected. The default value is calculated assuming the max clk_x time period of 4ns to work with ONFI Mode 0 mode of 100ns device cycle time. This assumes a 1x/5x clocking scheme.	RW	0x12

rdwr_en_hi_cnt

Read/Write Enable high pulse width

Module Instance	Base Address	Register Address
i_nand_config	0xFFB80000	0xFFB80200

Offset: 0x200

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved											value RW 0xC				

rdwr_en_hi_cnt Fields

Bit	Name	Description	Access	Reset
4:0	value	Number of clk_x cycles that read or write enable will kept high to meet the min Treh/Tweh parameter of the device. The value in this register plus rdwr_en_lo_cnt register value should meet the min cycle time of the device connected. The default value is calculated assuming the max clk_x time period of 4ns to work with ONFI Mode 0 mode of 100ns device cycle time. This assumes a 1x/5x clocking scheme.	RW	0xC

max_rd_delay

Max round trip read data delay for data capture

Module Instance	Base Address	Register Address
i_nand_config	0xFFB80000	0xFFB80210

Offset: 0x210

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												value RW 0x0			

max_rd_delay Fields

Bit	Name	Description	Access	Reset
3:0	value	Number of clk_x cycles after generation of feedback clk_x_out pulse when it is safe to synchronize received data to clk_x domain. Data should have been registered with clk_x_in and stable by the time max_rd_delay cycles has elapsed. Please see timing diagram in bus interface timing section of this guide for further elaboration. A default value of zero will mean a value of clk_x multiple minus one.	RW	0x0

cs_setup_cnt

Chip select setup/tWB time

Module Instance	Base Address	Register Address
i_nand_config	0xFFB80000	0xFFB80220

Offset: 0x220

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved														twb RW 0xA	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
twb RW 0xA				Reserved							value RW 0x3				

cs_setup_cnt Fields

Bit	Name	Description	Access	Reset
17:12	twb	Number of clk_x cycles required for meeting the tWB time. This register refers to device timing parameter TWB.	RW	0xA
4:0	value	Number of clk_x cycles required for meeting chip select setup time. This register refers to the device timing parameter Tcs. The value in this register reflects the extra setup cycles for chip select before read/write enable signal is set low. The default value is calculated for ONFI Timing mode 0 Tcs = 70ns and maximum clk_x period of 4ns for 1x/5x clock multiple for 20ns cycle time device.	RW	0x3

spare_area_skip_bytes

Spare area skip bytes

Module Instance	Base Address	Register Address
i_nand_config	0xFFB80000	0xFFB80230

Offset: 0x230

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										value RW 0x0					

spare_area_skip_bytes Fields

Bit	Name	Description	Access	Reset
5:0	value	Number of bytes to skip from start of spare area before last ECC sector data starts. The bytes will be written with the value programmed in the spare_area_marker register. This register could be potentially used to preserve the bad block marker in the spare area by marking it good. The default value is zero which means no bytes will be skipped and last ECC sector will start from the beginning of spare area. This value should be an even number.	RW	0x0

spare_area_marker

Spare area marker value

Module Instance	Base Address	Register Address
i_nand_config	0xFFB80000	0xFFB80240

Offset: 0x240

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
value RW 0xFFFF															

spare_area_marker Fields

Bit	Name	Description	Access	Reset
15:0	value	A 16bit value that will be written in the spare area skip bytes. This value will be used by controller while in the MAIN mode of data transfer.	RW	0xFFFF

devices_connected

Number of Devices connected on one bank

Module Instance	Base Address	Register Address
i_nand_config	0xFFB80000	0xFFB80250

Offset: 0x250

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													value RW 0x0		

devices_connected Fields

Bit	Name	Description	Access	Reset
2:0	value	Indicates the number of devices connected to a bank. At POR, the value loaded is the maximum possible devices that could be connected in this configuration.	RW	0x0

die_mask

Indicates the die differentiator in case of NAND devices with stacked dies.

Module Instance	Base Address	Register Address
i_nand_config	0xFFB80000	0xFFB80260

Offset: 0x260

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
value RW 0x0															

die_mask Fields

Bit	Name	Description	Access	Reset
15:0	value	The die_mask register information will be used for devices having address restrictions. For example, in certain Samsung devices, when the first address in a two-plane command is being sent, it is expected that the address is all zeros. But if the NAND device internally has multiple dies stacked, the die information (MSB of final row address) has to be sent. The value programmed in this register will be used to mask the address while sending out the last row address.	RW	0x0

first_block_of_next_plane

The starting block address of the next plane in a multi plane device.

Module Instance	Base Address	Register Address
i_nand_config	0xFFB80000	0xFFB80270

Offset: 0x270

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
value RW 0x1															

first_block_of_next_plane Fields

Bit	Name	Description	Access	Reset
15:0	value	This values informs the controller of the plane structure of the device. In case the device is a multi plane device and the value here is 1, the controller understands that the next plane starts from Block number 1 and in conjunction with the number of planes parameter can decide upon the distribution of blocks in a plane in the device.	RW	0x1

write_protect

This register is used to control the assertion/de-assertion of the WP# pin to the device.

Module Instance	Base Address	Register Address
i_nand_config	0xFFB80000	0xFFB80280

Offset: 0x280

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															flag RW 0x1

write_protect Fields

Bit	Name	Description	Access	Reset
0	flag	When the controller is in reset, the WP# pin is always asserted to the device. Once the reset is removed, the WP# is de-asserted. The software will then have to come and program this bit to assert/de-assert the same. [list][*]1 - Write protect de-assert [*]0 - Write protect assert[/list]	RW	0x1

re_2_re

Timing parameter between re high to re low (Trhz) for the next bank

Module Instance	Base Address	Register Address
i_nand_config	0xFFB80000	0xFFB80290

Offset: 0x290

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										value RW 0x32					

re_2_re Fields

Bit	Name	Description	Access	Reset
5:0	value	Signifies the number of bus interface clk_x clocks that should be introduced between read enable going high to a bank to the read enable going low to the next bank. The number of clocks is the function of device parameter Trhz and controller clock frequency.	RW	0x32

por_reset_count

The number of cycles the controller waits after POR to issue the first RESET command to the device.

Module Instance	Base Address	Register Address
i_nand_config	0xFFB80000	0xFFB802A0

Offset: 0x2A0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
value RW 0x13B															

por_reset_count Fields

Bit	Name	Description	Access	Reset
15:0	value	The controller waits for this number of cycles before issuing the first RESET command to the device. The number in this register is multiplied internally by 16 in the controller to form the final reset wait count.	RW	0x13B

watchdog_reset_count

The number of cycles the controller waits before flagging a watchdog timeout interrupt.

Module Instance	Base Address	Register Address
i_nand_config	0xFFB80000	0xFFB802B0

Offset: 0x2B0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
value RW 0x5B9A															

watchdog_reset_count Fields

Bit	Name	Description	Access	Reset
15:0	value	The controller waits for this number of cycles before issuing a watchdog timeout interrupt. The value in this register is multiplied internally by 32 in the controller to form the final watchdog counter.	RW	0x5B9A

device_reset

Device reset. Controller sends a RESET command to device. Controller resets bit after sending command to device

Module Instance	Base Address	Register Address
i_nand_config	0xFFB80000	0xFFB80000

Offset: 0x0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												bank3 RW 0x0	bank2 RW 0x0	bank1 RW 0x0	bank0 RW 0x0

device_reset Fields

Bit	Name	Description	Access	Reset
3	bank3	Issues reset to bank 3. Controller resets the bit after reset command is issued to device.	RW	0x0
2	bank2	Issues reset to bank 2. Controller resets the bit after reset command is issued to device.	RW	0x0
1	bank1	Issues reset to bank 1. Controller resets the bit after reset command is issued to device.	RW	0x0
0	bank0	Issues reset to bank 0. Controller resets the bit after reset command is issued to device.	RW	0x0

nand_param Address Map

Module Instance	Base Address	End Address
i_nand_param	0xFFB80300	0xFFB803FF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
manufacturer_id on page 13-195	0x0	32	RW	0x0	
device_id on page 13-196	0x10	32	RO	0x0	
device_param_0 on page 13-197	0x20	32	RO	0x0	
device_param_1 on page 13-197	0x30	32	RO	0x0	
device_param_2 on page 13-198	0x40	32	RO	0x0	
logical_page_data_size on page 13-199	0x50	32	RO	0x0	Logical page data area size in bytes
logical_page_spare_size on page 13-200	0x60	32	RO	0x0	Logical page data area size in bytes
revision on page 13-200	0x70	32	RO	0x105	Controller Revision
onfi_device_features on page 13-201	0x80	32	RO	0x0	Features supported by the connected ONFI device
onfi_optional_commands on page 13-203	0x90	32	RO	0x0	Optional commands supported by the connected ONFI device
onfi_timing_mode on page 13-205	0xA0	32	RO	0x0	Asynchronous Timing modes supported by the connected ONFI device
onfi_pgm_cache_timing_mode on page 13-206	0xB0	32	RO	0x0	Asynchronous Program Cache Timing modes supported by the connected ONFI device
onfi_device_no_of_luns on page 13-207	0xC0	32	RW	0x0	Indicates if the device is an ONFI compliant device and the number of LUNS present in the device

Register	Offset	Width	Access	Reset Value	Description
onfi_device_no_of_blocks_per_lun_l on page 13-209	0xD0	32	RO	0x0	Lower bits of number of blocks per LUN present in the ONFI complaint device.
onfi_device_no_of_blocks_per_lun_u on page 13-210	0xE0	32	RO	0x0	Upper bits of number of blocks per LUN present in the ONFI complaint device.
features on page 13-211	0xF0	32	RO	0x8C2	Shows Available hardware features or attributes

nand_param Summary

Base Address: 0xFFB80300

Register Address Offset	Bit Fields															
i_nand_param																
manufac-turer_id 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								value RW 0x0							
device_id 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								value RO 0x0							

Register Address Offset	Bit Fields															
device_param_0 0x20	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								value RO 0x0							
device_param_1 0x30	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								value RO 0x0							
device_param_2 0x40	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								value RO 0x0							
logical_page_data_size 0x50	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	value RO 0x0															
logical_page_spare_size 0x60	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	value RO 0x0															
revision 0x70	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	minor RO 0x1								value RO 0x5							
onfi_device_features 0x80	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	value RO 0x0															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
onfi_optional_commands 0x90	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	value RO 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
onfi_timing_mode 0xA0	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved										value RO 0x0					
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
onfi_pgm_cache_timing_mode 0xB0	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved										value RO 0x0					
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
onfi_device_no_of_luns 0xC0	Reserved											ce_reduction_volume_addr_and_change RW 0x0	Reserved			onfi_jedec_multiplane_erase_seq RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved			prog_page_reg_clear_enhancement RW 0x0	Reserved			onfi_device RW 0x0	no_of_luns RO 0x0							

Register Address Offset	Bit Fields															
onfi_device_no_of_blocks_per_lun_l 0xD0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	value RO 0x0															
onfi_device_no_of_blocks_per_lun_u 0xE0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	value RO 0x0															
features 0xF0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		lba RO 0x0	dfi_intf RO 0x0	index_addr RO 0x1	gpreg RO 0x0	xdma_sideband RO 0x0	partition RO 0x0	cmd_dma RO 0x1	dma RO 0x1	Reserved				n_banks RO 0x2	

manufacturer_id

Module Instance	Base Address	Register Address
i_nand_param	0xFFB80300	0xFFB80300

Offset: 0x0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								value RW 0x0							

manufacturer_id Fields

Bit	Name	Description	Access	Reset
7:0	value	Manufacturer ID	RW	0x0

device_id

Module Instance	Base Address	Register Address
i_nand_param	0xFFB80300	0xFFB80310

Offset: 0x10

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								value RO 0x0							

device_id Fields

Bit	Name	Description	Access	Reset
7:0	value	Device ID	RO	0x0

device_param_0

Module Instance	Base Address	Register Address
i_nand_param	0xFFB80300	0xFFB80320

Offset: 0x20

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								value RO 0x0							

device_param_0 Fields

Bit	Name	Description	Access	Reset
7:0	value	3rd byte relating to Device Signature. This register is updated only for Legacy NAND devices.	RO	0x0

device_param_1

Module Instance	Base Address	Register Address
i_nand_param	0xFFB80300	0xFFB80330

Offset: 0x30

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								value RO 0x0							

device_param_1 Fields

Bit	Name	Description	Access	Reset
7:0	value	4th byte relating to Device Signature. This register is updated only for Legacy NAND devices.	RO	0x0

device_param_2

Module Instance	Base Address	Register Address
i_nand_param	0xFFB80300	0xFFB80340

Offset: 0x40

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								value RO 0x0							

device_param_2 Fields

Bit	Name	Description	Access	Reset
7:0	value	Reserved.	RO	0x0

logical_page_data_size

Logical page data area size in bytes

Module Instance	Base Address	Register Address
i_nand_param	0xFFB80300	0xFFB80350

Offset: 0x50

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
value RO 0x0															

logical_page_data_size Fields

Bit	Name	Description	Access	Reset
15:0	value	Logical page spare area size in bytes. If multiple devices are connected on a single chip select, physical page data size will be multiplied by the number of devices to arrive at logical page size.	RO	0x0

logical_page_spare_size

Logical page data area size in bytes

Module Instance	Base Address	Register Address
i_nand_param	0xFFB80300	0xFFB80360

Offset: 0x60

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
value RO 0x0															

logical_page_spare_size Fields

Bit	Name	Description	Access	Reset
15:0	value	Logical page spare area size in bytes. If multiple devices are connected on a single chip select, physical page spare size will be multiplied by the number of devices to arrive at logical page size.	RO	0x0

revision

Controller Revision

Module Instance	Base Address	Register Address
i_nand_param	0xFFB80300	0xFFB80370

Offset: 0x70

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
minor RO 0x1								value RO 0x5							

revision Fields

Bit	Name	Description	Access	Reset
15:8	minor	The Minor revision number of the controller	RO	0x1
7:0	value	The Major revision number of the controller	RO	0x5

onfi_device_features

Features supported by the connected ONFI device

Module Instance	Base Address	Register Address
i_nand_param	0xFFB80300	0xFFB80380

Offset: 0x80

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
value RO 0x0															

onfi_device_features Fields

Bit	Name	Description	Access	Reset
15:0	value	The values in the field should be interpreted as follows[list] [*]Bit 0 - Supports 16 bit data bus width. [*]Bit 1 - Supports multiple LUN operations. [*]Bit 2 - Supports non-sequential page programming. [*]Bit 3 - Supports interleaved program and erase operations. [*]Bit 4 - Supports odd to even page copyback. [*]Bit 5 - Supports source synchro-nous. [*]Bit 6 - Supports interleaved read operations. [*]Bit 7 - Supports extended parameter page. [*]Bit 8 - Supports program page register clear enhancement. [*]Bit 9 - Supports EZNAND. [*]Bit 10 - Supports NV-DDR2. [*]Bit 11 - Supports Volume Addressing. [*]Bit 12 - Supports External Vpp. [*]Bit 13-15 - Reserved.[/list]	RO	0x0

onfi_optional_commands

Optional commands supported by the connected ONFI device

Module Instance	Base Address	Register Address
i_nand_param	0xFFB80300	0xFFB80390

Offset: 0x90

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
value RO 0x0															

onfi_optional_commands Fields

Bit	Name	Description	Access	Reset
15:0	value	The values in the field should be interpreted as follows[list] [*]Bit 0 - Supports page cache program command. [*]Bit 1 - Supports read cache commands. [*]Bit 2 - Supports get and set features. [*]Bit 3 - Supports read status enhanced commands. [*]Bit 4 - Supports copyback. [*]Bit 5 - Supports Read Unique Id. [*]Bit 6 - Supports Change Read Column Enhanced. [*]Bit 7 - Supports change row address. [*]Bit 8 - Supports Change small data move. [*]Bit 9 - Supports RESET LUN. [*]Bit 10 - Supports Volume Select. [*]Bit 11 - Supports ODT Configure. [*]Bit 12-15 - Reserved.[/list]	RO	0x0

onfi_timing_mode

Asynchronous Timing modes supported by the connected ONFI device

Module Instance	Base Address	Register Address
i_nand_param	0xFFB80300	0xFFB803A0

Offset: 0xA0

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										value RO 0x0					

onfi_timing_mode Fields

Bit	Name	Description	Access	Reset
5:0	value	The values in the field should be interpreted as follows[list] [*]Bit 0 - Supports Timing mode 0. [*]Bit 1 - Supports Timing mode 1. [*]Bit 2 - Supports Timing mode 2. [*]Bit 3 - Supports Timing mode 3. [*]Bit 4 - Supports Timing mode 4. [*]Bit 5 - Supports Timing mode 5.[/list]	RO	0x0

onfi_pgm_cache_timing_mode

Asynchronous Program Cache Timing modes supported by the connected ONFI device

Module Instance	Base Address	Register Address
i_nand_param	0xFFB80300	0xFFB803B0

Offset: 0xB0

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										value RO 0x0					

onfi_pgm_cache_timing_mode Fields

Bit	Name	Description	Access	Reset
5:0	value	The values in the field should be interpreted as follows[list] [*]Bit 0 - Supports Timing mode 0. [*]Bit 1 - Supports Timing mode 1. [*]Bit 2 - Supports Timing mode 2. [*]Bit 3 - Supports Timing mode 3. [*]Bit 4 - Supports Timing mode 4. [*]Bit 5 - Supports Timing mode 5.[/list]	RO	0x0

onfi_device_no_of_luns

Indicates if the device is an ONFI compliant device and the number of LUNS present in the device

Module Instance	Base Address	Register Address
i_nand_param	0xFFB80300	0xFFB803C0

Offset: 0xC0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved											ce_reduction_volume_addr_and_change RW 0x0	Reserved			onfi_jedec_multiplane_erase_seq RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			prog_page_reg_clear_enhancement RW 0x0	Reserved			onfi_device RW 0x0	no_of_luns RO 0x0							

onfi_device_no_of_luns Fields

Bit	Name	Description	Access	Reset
20	ce_reduction_volume_addr_and_change	Device supports CE pin reduction with volume assignments,volume addressing and volume change command sequence.For any device configured by host to be used in volume addressing mode, this bit must be set to 1. [list][*]1 - supported [*]0 - Not supported[/list]	RW	0x0

Bit	Name	Description	Access	Reset
16	onfi_jedec_multiplane_erase_seq	Device supports ONFI JEDEC Multiplane erase sequence.(Only valid for Onfi devices) [list][*]1 - ONFI JEDEC Multiplane erase sequence supported [*]0 - ONFI JEDEC Multiplane erase sequence not supported[/list]	RW	0x0
12	prog_page_reg_clear_enhancement	Device supports program page register clear enhancement.In such a device, a program can be initiated in a LUN even if a read command is ongoing in another LUN in the device. [list][*]1 - Program page register clear enhancement supported [*]0 - Program page register clear enhancement not supported[/list]	RW	0x0
8	onfi_device	Indicates if the device is an ONFI compliant device.[list] [*]0 - Non-ONFI compliant device [*]1 - ONFI compliant device[/list]	RW	0x0
7:0	no_of_luns	Indicates the number of LUNS present in the device	RO	0x0

onfi_device_no_of_blocks_per_lun_l

Lower bits of number of blocks per LUN present in the ONFI complaint device.

Module Instance	Base Address	Register Address
i_nand_param	0xFFB80300	0xFFB803D0

Offset: 0xD0

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
value RO 0x0															

onfi_device_no_of_blocks_per_lun_l Fields

Bit	Name	Description	Access	Reset
15:0	value	Indicates the lower bits of number of blocks per LUN present in the ONFI complaint device.	RO	0x0

onfi_device_no_of_blocks_per_lun_u

Upper bits of number of blocks per LUN present in the ONFI complaint device.

Module Instance	Base Address	Register Address
i_nand_param	0xFFB80300	0xFFB803E0

Offset: 0xE0

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
value RO 0x0															

onfi_device_no_of_blocks_per_lun_u Fields

Bit	Name	Description	Access	Reset
15:0	value	Indicates the upper bits of number of blocks per LUN present in the ONFI complaint device.	RO	0x0

features

Shows Available hardware features or attributes

Module Instance	Base Address	Register Address
i_nand_param	0xFFB80300	0xFFB803F0

Offset: 0xF0

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		lba RO 0x0	dfi_intf RO 0x0	index_addr RO 0x1	gpreg RO 0x0	xdma_sideband RO 0x0	partition RO 0x0	cmd_dma RO 0x1	dma RO 0x1	Reserved				n_banks RO 0x2	

features Fields

Bit	Name	Description	Access	Reset
13	lba	if set, hardware supports Toshiba LBA devices.	RO	0x0
12	dfi_intf	if set, hardware supports ONFI2.x synchronous interface.	RO	0x0
11	index_addr	if set, hardware support only Indexed addressing.	RO	0x1
10	gpreg	if set, General purpose registers are is present in hardware.	RO	0x0
9	xdma_sideband	if set, Side band DMA signals are present in hardware.	RO	0x0
8	partition	if set, Partition logic is present in hardware.	RO	0x0
7	cmd_dma	if set, CMD-DMA is present in hardware.	RO	0x1
6	dma	if set, DATA-DMA is present in hardware.	RO	0x1
1:0	n_banks	Maximum number of banks supported by hardware. This is an encoded value. [list][*]0 - One bank [*]1 - Two banks [*]2 - Four banks [*]3 - Eight banks[/list]	RO	0x2

nand_status Address Map

Module Instance	Base Address	End Address
i_nand_status	0xFFB80400	0xFFB8064F

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
transfer_mode on page 13-219	0x0	32	RO	0x0	Current data transfer mode is Main only, Spare only or Main +Spare. This information is per bank.
intr_status0 on page 13-220	0x10	32	RW	0x0	Interrupt status register for bank 0
intr_en0 on page 13-223	0x20	32	RW	0x2000	Enables corresponding interrupt bit in interrupt register for bank 0
page_cnt0 on page 13-226	0x30	32	RO	0x0	Decrementing page count bank 0
err_page_addr0 on page 13-226	0x40	32	RO	0x0	Erred page address bank 0
err_block_addr0 on page 13-227	0x50	32	RO	0x0	Erred block address bank 0
intr_status1 on page 13-228	0x60	32	RW	0x0	Interrupt status register for bank 1
intr_en1 on page 13-231	0x70	32	RW	0x2000	Enables corresponding interrupt bit in interrupt register for bank 1
page_cnt1 on page 13-234	0x80	32	RO	0x0	Decrementing page count bank 1
err_page_addr1 on page 13-234	0x90	32	RO	0x0	Erred page address bank 1
err_block_addr1 on page 13-235	0xA0	32	RO	0x0	Erred block address bank 1

Register	Offset	Width	Access	Reset Value	Description
intr_status2 on page 13-236	0xB0	32	RW	0x0	Interrupt status register for bank 2
intr_en2 on page 13-239	0xC0	32	RW	0x2000	Enables corresponding interrupt bit in interrupt register for bank 2
page_cnt2 on page 13-242	0xD0	32	RO	0x0	Decrementing page count bank 2
err_page_addr2 on page 13-242	0xE0	32	RO	0x0	Erred page address bank 2
err_block_addr2 on page 13-243	0xF0	32	RO	0x0	Erred block address bank 2
intr_status3 on page 13-244	0x100	32	RW	0x0	Interrupt status register for bank 3
intr_en3 on page 13-247	0x110	32	RW	0x2000	Enables corresponding interrupt bit in interrupt register for bank 3
page_cnt3 on page 13-250	0x120	32	RO	0x0	Decrementing page count bank 3
err_page_addr3 on page 13-250	0x130	32	RO	0x0	Erred page address bank 3
err_block_addr3 on page 13-251	0x140	32	RO	0x0	Erred block address bank 3

nand_status Summary

Base Address: 0xFFB80400

Register Address Offset	Bit Fields
i_nand_status	

Register Address Offset	Bit Fields															
transfer_mode 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								value3 RO 0x0		value2 RO 0x0		value1 RO 0x0		value0 RO 0x0	
intr_status0 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															erased_page RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	page_xfer_inc RW 0x0	pipe_cmd_err RW 0x0	rst_comp RW 0x0	int_act RW 0x0	unsup_cmd RW 0x0	locked_blk RW 0x0	pipe_cpybck_cmd_comp RW 0x0	erase_comp RW 0x0	program_comp RW 0x0	load_comp RW 0x0	erase_fail RW 0x0	program_fail RW 0x0	time_out RW 0x0	dma_cmd_comp RW 0x0	Reserved	ecc_uncor_err RW 0x0
intr_en0 0x20	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															erased_page RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	page_xfer_inc RW 0x0	pipe_cmd_err RW 0x0	rst_comp RW 0x1	int_act RW 0x0	unsup_cmd RW 0x0	locked_blk RW 0x0	pipe_cpybck_cmd_comp RW 0x0	erase_comp RW 0x0	program_comp RW 0x0	load_comp RW 0x0	erase_fail RW 0x0	program_fail RW 0x0	time_out RW 0x0	dma_cmd_comp RW 0x0	Reserved	ecc_uncor_err RW 0x0
page_cnt0 0x30	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								value RO 0x0							
err_page_addr0 0x40	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	value RO 0x0															

Register Address Offset	Bit Fields															
err_block_addr0 0x50	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	value RO 0x0															
intr_status1 0x60	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															erased_page RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	page_xfer_inc RW 0x0	pipe_cmd_err RW 0x0	rst_comp RW 0x0	int_act RW 0x0	unsup_cmd RW 0x0	locked_blk RW 0x0	pipe_cpyback_cmd_comp RW 0x0	erase_comp RW 0x0	program_comp RW 0x0	load_comp RW 0x0	erase_fail RW 0x0	program_fail RW 0x0	time_out RW 0x0	dma_cmd_comp RW 0x0	Reserved	ecc_uncor_err RW 0x0
intr_en1 0x70	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															erased_page RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	page_xfer_inc RW 0x0	pipe_cmd_err RW 0x0	rst_comp RW 0x1	int_act RW 0x0	unsup_cmd RW 0x0	locked_blk RW 0x0	pipe_cpyback_cmd_comp RW 0x0	erase_comp RW 0x0	program_comp RW 0x0	load_comp RW 0x0	erase_fail RW 0x0	program_fail RW 0x0	time_out RW 0x0	dma_cmd_comp RW 0x0	Reserved	ecc_uncor_err RW 0x0
page_cnt1 0x80	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								value RO 0x0							
err_page_addr1 0x90	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	value RO 0x0															

Register Address Offset	Bit Fields															
err_block_addr1 0xA0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	value RO 0x0															
intr_status2 0xB0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															erased_page RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	page_xfer_inc RW 0x0	pipe_cmd_err RW 0x0	rst_comp RW 0x0	int_act RW 0x0	unsup_cmd RW 0x0	locked_blk RW 0x0	pipe_cpyback_cmd_comp RW 0x0	erase_comp RW 0x0	program_comp RW 0x0	load_comp RW 0x0	erase_fail RW 0x0	program_fail RW 0x0	time_out RW 0x0	dma_cmd_comp RW 0x0	Reserved	ecc_uncor_err RW 0x0
intr_en2 0xC0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															erased_page RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	page_xfer_inc RW 0x0	pipe_cmd_err RW 0x0	rst_comp RW 0x1	int_act RW 0x0	unsup_cmd RW 0x0	locked_blk RW 0x0	pipe_cpyback_cmd_comp RW 0x0	erase_comp RW 0x0	program_comp RW 0x0	load_comp RW 0x0	erase_fail RW 0x0	program_fail RW 0x0	time_out RW 0x0	dma_cmd_comp RW 0x0	Reserved	ecc_uncor_err RW 0x0
page_cnt2 0xD0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								value RO 0x0							
err_page_addr2 0xE0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	value RO 0x0															

Register Address Offset	Bit Fields															
err_block_addr2 0xF0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	value RO 0x0															
intr_status3 0x100	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															erased_page RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	page_xfer_inc RW 0x0	pipe_cmd_err RW 0x0	rst_comp RW 0x0	int_act RW 0x0	unsup_cmd RW 0x0	locked_blk RW 0x0	pipe_cpyback_cmd_comp RW 0x0	erase_comp RW 0x0	program_comp RW 0x0	load_comp RW 0x0	erase_fail RW 0x0	program_fail RW 0x0	time_out RW 0x0	dma_cmd_comp RW 0x0	Reserved	ecc_uncor_err RW 0x0
intr_en3 0x110	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															erased_page RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	page_xfer_inc RW 0x0	pipe_cmd_err RW 0x0	rst_comp RW 0x1	int_act RW 0x0	unsup_cmd RW 0x0	locked_blk RW 0x0	pipe_cpyback_cmd_comp RW 0x0	erase_comp RW 0x0	program_comp RW 0x0	load_comp RW 0x0	erase_fail RW 0x0	program_fail RW 0x0	time_out RW 0x0	dma_cmd_comp RW 0x0	Reserved	ecc_uncor_err RW 0x0
page_cnt3 0x120	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								value RO 0x0							
err_page_addr3 0x130	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	value RO 0x0															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
err_block_addr3 0x140	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	value RO 0x0															

transfer_mode

Current data transfer mode is Main only, Spare only or Main +Spare.

This information is per bank.

Module Instance	Base Address	Register Address
i_nand_status	0xFFB80400	0xFFB80400

Offset: 0x0

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								value3 RO 0x0		value2 RO 0x0		value1 RO 0x0		value0 RO 0x0	

transfer_mode Fields

Bit	Name	Description	Access	Reset
7:6	value3	[list][*]00 - Bank 3 is in Main mode [*]01 - Bank 3 is in Spare mode [*]10 - Bank 3 is in Main +Spare mode[/list]	RO	0x0

Bit	Name	Description	Access	Reset
5:4	value2	[list][*]00 - Bank 2 is in Main mode [*]01 - Bank 2 is in Spare mode [*]10 - Bank 2 is in Main +Spare mode[/list]	RO	0x0
3:2	value1	[list][*]00 - Bank 1 is in Main mode [*]01 - Bank 1 is in Spare mode [*]10 - Bank 1 is in Main +Spare mode[/list]	RO	0x0
1:0	value0	[list][*]00 - Bank 0 is in Main mode [*]01 - Bank 0 is in Spare mode [*]10 - Bank 0 is in Main +Spare mode[/list]	RO	0x0

intr_status0

Interrupt status register for bank 0

Module Instance	Base Address	Register Address
i_nand_status	0xFFB80400	0xFFB80410

Offset: 0x10

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															erased_page RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
page_xfer_inc RW 0x0	pipe_cmd_err RW 0x0	rst_comp RW 0x0	int_act RW 0x0	unsup_cmd RW 0x0	locked_blk RW 0x0	pipe_cpybck_cmd_comp RW 0x0	erase_comp RW 0x0	program_comp RW 0x0	load_comp RW 0x0	erase_fail RW 0x0	program_fail RW 0x0	time_out RW 0x0	dma_cmd_comp RW 0x0	Reserved	ecc_uncor_err RW 0x0

intr_status0 Fields

Bit	Name	Description	Access	Reset
16	erased_page	If an erased page is detected on reads, this bit will be set. The detection of erased page is based on the number of 0's in the page. If the number of 0's in the page being read is less than the value in the erase_threshold (programmable register), an erased page is inferred and no un-correctable error will be flagged for that page. If ECC is disabled, the erased_page interrupt shall be set as explained above. If ECC is enabled, in addition to the above condition, only when the ECC logic detects an un-correctable error for that page will the erased_page interrupt be flagged. If the ECC logic detects a no-error or correctable error page, this erased page interrupt will not be set.	RW	0x0
15	page_xfer_inc	For every page of data transfer to or from the device, this bit will be set.	RW	0x0
14	pipe_cmd_err	A pipeline command sequence has been violated. This occurs when Map 01 page read/write address does not match the corresponding expected address from the pipeline commands issued earlier.	RW	0x0
13	rst_comp	Controller has finished reset and initialization process	RW	0x0
12	int_act	R/B pin of device transitioned from low to high	RW	0x0

Bit	Name	Description	Access	Reset
11	unsup_cmd	An unsupported command was received. This interrupt is set when an invalid command is received, or when a command sequence is broken.	RW	0x0
10	locked_blk	The address to program or erase operation is to a locked block and the operation failed due to this reason	RW	0x0
9	pipe_cpybck_cmd_comp	A pipeline command or a copyback bank command has completed on this particular bank	RW	0x0
8	erase_comp	Device erase operation complete	RW	0x0
7	program_comp	Device finished the last issued program command.	RW	0x0
6	load_comp	Device finished the last issued load command.	RW	0x0
5	erase_fail	Erase failure occurred in the device on issuance of a erase command. err_block_addr and err_page_addr contain the block address and page address that failed erase operation.	RW	0x0
4	program_fail	Program failure occurred in the device on issuance of a program command. err_block_addr and err_page_addr contain the block address and page address that failed program operation.	RW	0x0
3	time_out	Watchdog timer has triggered in the controller due to one of the reasons like device not responding or controller state machine did not get back to idle	RW	0x0
2	dma_cmd_comp	A data DMA command has completed on this bank	RW	0x0

Bit	Name	Description	Access	Reset
0	ecc_uncor_err	Ecc logic detected uncorrectable error while reading data from flash device.	RW	0x0

intr_en0

Enables corresponding interrupt bit in interrupt register for bank 0

Module Instance	Base Address	Register Address
i_nand_status	0xFFB80400	0xFFB80420

Offset: 0x20

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															erased_page RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
page_xfer_inc RW 0x0	pipe_cmd_err RW 0x0	rst_comp RW 0x1	int_act RW 0x0	unsup_cmd RW 0x0	locked_blk RW 0x0	pipe_cpybck_cmd_comp RW 0x0	erase_comp RW 0x0	program_comp RW 0x0	load_comp RW 0x0	erase_fail RW 0x0	program_fail RW 0x0	time_out RW 0x0	dma_cmd_comp RW 0x0	Reserved	ecc_uncor_err RW 0x0

intr_en0 Fields

Bit	Name	Description	Access	Reset
16	erased_page	If an erased page is detected on reads, this bit will be set. The detection of erased page is based on the number of 0's in the page. If the number of 0's in the page being read is less than the value in the erase_threshold (programmable register), an erased page is inferred and no un-correctable error will be flagged for that page. If ECC is disabled, the erased_page interrupt shall be set as explained above. If ECC is enabled, in addition to the above condition, only when the ECC logic detects an un-correctable error for that page will the erased_page interrupt be flagged. If the ECC logic detects a no-error or correctable error page, this erased page interrupt will not be set.	RW	0x0
15	page_xfer_inc	For every page of data transfer to or from the device, this bit will be set.	RW	0x0
14	pipe_cmd_err	A pipeline command sequence has been violated. This occurs when Map 01 page read/write address does not match the corresponding expected address from the pipeline commands issued earlier.	RW	0x0
13	rst_comp	A reset command has completed on this bank	RW	0x1
12	int_act	R/B pin of device transitioned from low to high	RW	0x0

Bit	Name	Description	Access	Reset
11	unsup_cmd	An unsupported command was received. This interrupt is set when an invalid command is received, or when a command sequence is broken.	RW	0x0
10	locked_blk	The address to program or erase operation is to a locked block and the operation failed due to this reason	RW	0x0
9	pipe_cpybck_cmd_comp	A pipeline command or a copyback bank command has completed on this particular bank	RW	0x0
8	erase_comp	Device erase operation complete	RW	0x0
7	program_comp	Device finished the last issued program command.	RW	0x0
6	load_comp	Device finished the last issued load command.	RW	0x0
5	erase_fail	Erase failure occurred in the device on issuance of a erase command. err_block_addr and err_page_addr contain the block address and page address that failed erase operation.	RW	0x0
4	program_fail	Program failure occurred in the device on issuance of a program command. err_block_addr and err_page_addr contain the block address and page address that failed program operation.	RW	0x0
3	time_out	Watchdog timer has triggered in the controller due to one of the reasons like device not responding or controller state machine did not get back to idle	RW	0x0
2	dma_cmd_comp	A data DMA command has completed on this bank	RW	0x0

Bit	Name	Description	Access	Reset
0	ecc_uncor_err	If set, Controller will interrupt processor when Ecc logic detects uncorrectable error.	RW	0x0

page_cnt0

Decrementing page count bank 0

Module Instance	Base Address	Register Address
i_nand_status	0xFFB80400	0xFFB80430

Offset: 0x30

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								value RO 0x0							

page_cnt0 Fields

Bit	Name	Description	Access	Reset
7:0	value	Maintains a decrementing count of the number of pages in the multi-page (pipeline and copyback) command being executed.	RO	0x0

err_page_addr0

Erred page address bank 0

Module Instance	Base Address	Register Address
i_nand_status	0xFFB80400	0xFFB80440

Offset: 0x40

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
value RO 0x0															

err_page_addr0 Fields

Bit	Name	Description	Access	Reset
15:0	value	Holds the page address that resulted in a failure on program or erase operation.	RO	0x0

err_block_addr0

Erred block address bank 0

Module Instance	Base Address	Register Address
i_nand_status	0xFFB80400	0xFFB80450

Offset: 0x50

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
value RO 0x0															

err_block_addr0 Fields

Bit	Name	Description	Access	Reset
15:0	value	Holds the block address that resulted in a failure on program or erase operation.	RO	0x0

intr_status1

Interrupt status register for bank 1

Module Instance	Base Address	Register Address
i_nand_status	0xFFB80400	0xFFB80460

Offset: 0x60

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															erased_page RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
page_xfer_inc RW 0x0	pipe_cmd_err RW 0x0	rst_comp RW 0x0	int_act RW 0x0	unsup_cmd RW 0x0	locked_blk RW 0x0	pipe_cpybck_cmd_comp RW 0x0	erase_comp RW 0x0	program_comp RW 0x0	load_comp RW 0x0	erase_fail RW 0x0	program_fail RW 0x0	time_out RW 0x0	dma_cmd_comp RW 0x0	Reserved	ecc_uncor_err RW 0x0

intr_status1 Fields

Bit	Name	Description	Access	Reset
16	erased_page	If an erased page is detected on reads, this bit will be set. The detection of erased page is based on the number of 0's in the page. If the number of 0's in the page being read is less than the value in the erase_threshold (programmable register), an erased page is inferred and no un-correctable error will be flagged for that page. If ECC is disabled, the erased_page interrupt shall be set as explained above. If ECC is enabled, in addition to the above condition, only when the ECC logic detects an un-correctable error for that page will the erased_page interrupt be flagged. If the ECC logic detects a no-error or correctable error page, this erased page interrupt will not be set.	RW	0x0
15	page_xfer_inc	For every page of data transfer to or from the device, this bit will be set.	RW	0x0
14	pipe_cmd_err	A pipeline command sequence has been violated. This occurs when Map 01 page read/write address does not match the corresponding expected address from the pipeline commands issued earlier.	RW	0x0
13	rst_comp	The Cadence NAND Flash Memory Controller has completed its reset and initialization process	RW	0x0
12	int_act	R/B pin of device transitioned from low to high	RW	0x0

Bit	Name	Description	Access	Reset
11	unsup_cmd	An unsupported command was received. This interrupt is set when an invalid command is received, or when a command sequence is broken.	RW	0x0
10	locked_blk	The address to program or erase operation is to a locked block and the operation failed due to this reason	RW	0x0
9	pipe_cpybck_cmd_comp	A pipeline command or a copyback bank command has completed on this particular bank	RW	0x0
8	erase_comp	Device erase operation complete	RW	0x0
7	program_comp	Device finished the last issued program command.	RW	0x0
6	load_comp	Device finished the last issued load command.	RW	0x0
5	erase_fail	Erase failure occurred in the device on issuance of a erase command. err_block_addr and err_page_addr contain the block address and page address that failed erase operation.	RW	0x0
4	program_fail	Program failure occurred in the device on issuance of a program command. err_block_addr and err_page_addr contain the block address and page address that failed program operation.	RW	0x0
3	time_out	Watchdog timer has triggered in the controller due to one of the reasons like device not responding or controller state machine did not get back to idle	RW	0x0
2	dma_cmd_comp	A data DMA command has completed on this bank	RW	0x0

Bit	Name	Description	Access	Reset
0	ecc_uncor_err	Ecc logic detected uncorrectable error while reading data from flash device.	RW	0x0

intr_en1

Enables corresponding interrupt bit in interrupt register for bank 1

Module Instance	Base Address	Register Address
i_nand_status	0xFFB80400	0xFFB80470

Offset: 0x70

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															erased_page RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
page_xfer_inc RW 0x0	pipe_cmd_err RW 0x0	rst_comp RW 0x1	int_act RW 0x0	unsup_cmd RW 0x0	locked_blk RW 0x0	pipe_cpybck_cmd_comp RW 0x0	erase_comp RW 0x0	program_comp RW 0x0	load_comp RW 0x0	erase_fail RW 0x0	program_fail RW 0x0	time_out RW 0x0	dma_cmd_comp RW 0x0	Reserved	ecc_uncor_err RW 0x0

intr_en1 Fields

Bit	Name	Description	Access	Reset
16	erased_page	If an erased page is detected on reads, this bit will be set. The detection of erased page is based on the number of 0's in the page. If the number of 0's in the page being read is less than the value in the erase_threshold (programmable register), an erased page is inferred and no un-correctable error will be flagged for that page. If ECC is disabled, the erased_page interrupt shall be set as explained above. If ECC is enabled, in addition to the above condition, only when the ECC logic detects an un-correctable error for that page will the erased_page interrupt be flagged. If the ECC logic detects a no-error or correctable error page, this erased page interrupt will not be set.	RW	0x0
15	page_xfer_inc	For every page of data transfer to or from the device, this bit will be set.	RW	0x0
14	pipe_cmd_err	A pipeline command sequence has been violated. This occurs when Map 01 page read/write address does not match the corresponding expected address from the pipeline commands issued earlier.	RW	0x0
13	rst_comp	A reset command has completed on this bank	RW	0x1
12	int_act	R/B pin of device transitioned from low to high	RW	0x0

Bit	Name	Description	Access	Reset
11	unsup_cmd	An unsupported command was received. This interrupt is set when an invalid command is received, or when a command sequence is broken.	RW	0x0
10	locked_blk	The address to program or erase operation is to a locked block and the operation failed due to this reason	RW	0x0
9	pipe_cpybck_cmd_comp	A pipeline command or a copyback bank command has completed on this particular bank	RW	0x0
8	erase_comp	Device erase operation complete	RW	0x0
7	program_comp	Device finished the last issued program command.	RW	0x0
6	load_comp	Device finished the last issued load command.	RW	0x0
5	erase_fail	Erase failure occurred in the device on issuance of a erase command. err_block_addr and err_page_addr contain the block address and page address that failed erase operation.	RW	0x0
4	program_fail	Program failure occurred in the device on issuance of a program command. err_block_addr and err_page_addr contain the block address and page address that failed program operation.	RW	0x0
3	time_out	Watchdog timer has triggered in the controller due to one of the reasons like device not responding or controller state machine did not get back to idle	RW	0x0
2	dma_cmd_comp	A data DMA command has completed on this bank	RW	0x0

Bit	Name	Description	Access	Reset
0	ecc_uncor_err	If set, Controller will interrupt processor when Ecc logic detects uncorrectable error.	RW	0x0

page_cnt1

Decrementing page count bank 1

Module Instance	Base Address	Register Address
i_nand_status	0xFFB80400	0xFFB80480

Offset: 0x80

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								value RO 0x0							

page_cnt1 Fields

Bit	Name	Description	Access	Reset
7:0	value	Maintains a decrementing count of the number of pages in the multi-page (pipeline and copyback) command being executed.	RO	0x0

err_page_addr1

Erred page address bank 1

Module Instance	Base Address	Register Address
i_nand_status	0xFFB80400	0xFFB80490

Offset: 0x90

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
value RO 0x0															

err_page_addr1 Fields

Bit	Name	Description	Access	Reset
15:0	value	Holds the page address that resulted in a failure on program or erase operation.	RO	0x0

err_block_addr1

Erred block address bank 1

Module Instance	Base Address	Register Address
i_nand_status	0xFFB80400	0xFFB804A0

Offset: 0xA0

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
value RO 0x0															

err_block_addr1 Fields

Bit	Name	Description	Access	Reset
15:0	value	Holds the block address that resulted in a failure on program or erase operation.	RO	0x0

intr_status2

Interrupt status register for bank 2

Module Instance	Base Address	Register Address
i_nand_status	0xFFB80400	0xFFB804B0

Offset: 0xB0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															erased_page RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
page_xfer_inc RW 0x0	pipe_cmd_err RW 0x0	rst_comp RW 0x0	int_act RW 0x0	unsup_cmd RW 0x0	locked_blk RW 0x0	pipe_cpybck_cmd_comp RW 0x0	erase_comp RW 0x0	program_comp RW 0x0	load_comp RW 0x0	erase_fail RW 0x0	program_fail RW 0x0	time_out RW 0x0	dma_cmd_comp RW 0x0	Reserved	ecc_uncor_err RW 0x0

intr_status2 Fields

Bit	Name	Description	Access	Reset
16	erased_page	If an erased page is detected on reads, this bit will be set. The detection of erased page is based on the number of 0's in the page. If the number of 0's in the page being read is less than the value in the erase_threshold (programmable register), an erased page is inferred and no un-correctable error will be flagged for that page. If ECC is disabled, the erased_page interrupt shall be set as explained above. If ECC is enabled, in addition to the above condition, only when the ECC logic detects an un-correctable error for that page will the erased_page interrupt be flagged. If the ECC logic detects a no-error or correctable error page, this erased page interrupt will not be set.	RW	0x0
15	page_xfer_inc	For every page of data transfer to or from the device, this bit will be set.	RW	0x0
14	pipe_cmd_err	A pipeline command sequence has been violated. This occurs when Map 01 page read/write address does not match the corresponding expected address from the pipeline commands issued earlier.	RW	0x0
13	rst_comp	The Cadence NAND Flash Memory Controller has completed its reset and initialization process	RW	0x0
12	int_act	R/B pin of device transitioned from low to high	RW	0x0

Bit	Name	Description	Access	Reset
11	unsup_cmd	An unsupported command was received. This interrupt is set when an invalid command is received, or when a command sequence is broken.	RW	0x0
10	locked_blk	The address to program or erase operation is to a locked block and the operation failed due to this reason	RW	0x0
9	pipe_cpybck_cmd_comp	A pipeline command or a copyback bank command has completed on this particular bank	RW	0x0
8	erase_comp	Device erase operation complete	RW	0x0
7	program_comp	Device finished the last issued program command.	RW	0x0
6	load_comp	Device finished the last issued load command.	RW	0x0
5	erase_fail	Erase failure occurred in the device on issuance of a erase command. err_block_addr and err_page_addr contain the block address and page address that failed erase operation.	RW	0x0
4	program_fail	Program failure occurred in the device on issuance of a program command. err_block_addr and err_page_addr contain the block address and page address that failed program operation.	RW	0x0
3	time_out	Watchdog timer has triggered in the controller due to one of the reasons like device not responding or controller state machine did not get back to idle	RW	0x0
2	dma_cmd_comp	A data DMA command has completed on this bank	RW	0x0

Bit	Name	Description	Access	Reset
0	ecc_uncor_err	Ecc logic detected uncorrectable error while reading data from flash device.	RW	0x0

intr_en2

Enables corresponding interrupt bit in interrupt register for bank 2

Module Instance	Base Address	Register Address
i_nand_status	0xFFB80400	0xFFB804C0

Offset: 0xC0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															erased_page RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
page_xfer_inc RW 0x0	pipe_cmd_err RW 0x0	rst_comp RW 0x1	int_act RW 0x0	unsup_cmd RW 0x0	locked_blk RW 0x0	pipe_cpybck_cmd_comp RW 0x0	erase_comp RW 0x0	program_comp RW 0x0	load_comp RW 0x0	erase_fail RW 0x0	program_fail RW 0x0	time_out RW 0x0	dma_cmd_comp RW 0x0	Reserved	ecc_uncor_err RW 0x0

intr_en2 Fields

Bit	Name	Description	Access	Reset
16	erased_page	If an erased page is detected on reads, this bit will be set. The detection of erased page is based on the number of 0's in the page. If the number of 0's in the page being read is less than the value in the erase_threshold (programmable register), an erased page is inferred and no un-correctable error will be flagged for that page. If ECC is disabled, the erased_page interrupt shall be set as explained above. If ECC is enabled, in addition to the above condition, only when the ECC logic detects an un-correctable error for that page will the erased_page interrupt be flagged. If the ECC logic detects a no-error or correctable error page, this erased page interrupt will not be set.	RW	0x0
15	page_xfer_inc	For every page of data transfer to or from the device, this bit will be set.	RW	0x0
14	pipe_cmd_err	A pipeline command sequence has been violated. This occurs when Map 01 page read/write address does not match the corresponding expected address from the pipeline commands issued earlier.	RW	0x0
13	rst_comp	A reset command has completed on this bank	RW	0x1
12	int_act	R/B pin of device transitioned from low to high	RW	0x0

Bit	Name	Description	Access	Reset
11	unsup_cmd	An unsupported command was received. This interrupt is set when an invalid command is received, or when a command sequence is broken.	RW	0x0
10	locked_blk	The address to program or erase operation is to a locked block and the operation failed due to this reason	RW	0x0
9	pipe_cpybck_cmd_comp	A pipeline command or a copyback bank command has completed on this particular bank	RW	0x0
8	erase_comp	Device erase operation complete	RW	0x0
7	program_comp	Device finished the last issued program command.	RW	0x0
6	load_comp	Device finished the last issued load command.	RW	0x0
5	erase_fail	Erase failure occurred in the device on issuance of a erase command. err_block_addr and err_page_addr contain the block address and page address that failed erase operation.	RW	0x0
4	program_fail	Program failure occurred in the device on issuance of a program command. err_block_addr and err_page_addr contain the block address and page address that failed program operation.	RW	0x0
3	time_out	Watchdog timer has triggered in the controller due to one of the reasons like device not responding or controller state machine did not get back to idle	RW	0x0
2	dma_cmd_comp	A data DMA command has completed on this bank	RW	0x0

Bit	Name	Description	Access	Reset
0	ecc_uncor_err	If set, Controller will interrupt processor when Ecc logic detects uncorrectable error.	RW	0x0

page_cnt2

Decrementing page count bank 2

Module Instance	Base Address	Register Address
i_nand_status	0xFFB80400	0xFFB804D0

Offset: 0xD0

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								value RO 0x0							

page_cnt2 Fields

Bit	Name	Description	Access	Reset
7:0	value	Maintains a decrementing count of the number of pages in the multi-page (pipeline and copyback) command being executed.	RO	0x0

err_page_addr2

Erred page address bank 2

Module Instance	Base Address	Register Address
i_nand_status	0xFFB80400	0xFFB804E0

Offset: 0xE0

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
value RO 0x0															

err_page_addr2 Fields

Bit	Name	Description	Access	Reset
15:0	value	Holds the page address that resulted in a failure on program or erase operation.	RO	0x0

err_block_addr2

Erred block address bank 2

Module Instance	Base Address	Register Address
i_nand_status	0xFFB80400	0xFFB804F0

Offset: 0xF0

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
value RO 0x0															

err_block_addr2 Fields

Bit	Name	Description	Access	Reset
15:0	value	Holds the block address that resulted in a failure on program or erase operation.	RO	0x0

intr_status3

Interrupt status register for bank 3

Module Instance	Base Address	Register Address
i_nand_status	0xFFB80400	0xFFB80500

Offset: 0x100

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															erased_page RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
page_xfer_inc RW 0x0	pipe_cmd_err RW 0x0	rst_comp RW 0x0	int_act RW 0x0	unsup_cmd RW 0x0	locked_blk RW 0x0	pipe_cpybck_cmd_comp RW 0x0	erase_comp RW 0x0	program_comp RW 0x0	load_comp RW 0x0	erase_fail RW 0x0	program_fail RW 0x0	time_out RW 0x0	dma_cmd_comp RW 0x0	Reserved	ecc_uncor_err RW 0x0

intr_status3 Fields

Bit	Name	Description	Access	Reset
16	erased_page	If an erased page is detected on reads, this bit will be set. The detection of erased page is based on the number of 0's in the page. If the number of 0's in the page being read is less than the value in the erase_threshold (programmable register), an erased page is inferred and no un-correctable error will be flagged for that page. If ECC is disabled, the erased_page interrupt shall be set as explained above. If ECC is enabled, in addition to the above condition, only when the ECC logic detects an un-correctable error for that page will the erased_page interrupt be flagged. If the ECC logic detects a no-error or correctable error page, this erased page interrupt will not be set.	RW	0x0
15	page_xfer_inc	For every page of data transfer to or from the device, this bit will be set.	RW	0x0
14	pipe_cmd_err	A pipeline command sequence has been violated. This occurs when Map 01 page read/write address does not match the corresponding expected address from the pipeline commands issued earlier.	RW	0x0
13	rst_comp	The Cadence NAND Flash Memory Controller has completed its reset and initialization process	RW	0x0
12	int_act	R/B pin of device transitioned from low to high	RW	0x0

Bit	Name	Description	Access	Reset
11	unsup_cmd	An unsupported command was received. This interrupt is set when an invalid command is received, or when a command sequence is broken.	RW	0x0
10	locked_blk	The address to program or erase operation is to a locked block and the operation failed due to this reason	RW	0x0
9	pipe_cpybck_cmd_comp	A pipeline command or a copyback bank command has completed on this particular bank	RW	0x0
8	erase_comp	Device erase operation complete	RW	0x0
7	program_comp	Device finished the last issued program command.	RW	0x0
6	load_comp	Device finished the last issued load command.	RW	0x0
5	erase_fail	Erase failure occurred in the device on issuance of a erase command. err_block_addr and err_page_addr contain the block address and page address that failed erase operation.	RW	0x0
4	program_fail	Program failure occurred in the device on issuance of a program command. err_block_addr and err_page_addr contain the block address and page address that failed program operation.	RW	0x0
3	time_out	Watchdog timer has triggered in the controller due to one of the reasons like device not responding or controller state machine did not get back to idle	RW	0x0
2	dma_cmd_comp	A data DMA command has completed on this bank	RW	0x0

Bit	Name	Description	Access	Reset
0	ecc_uncor_err	Ecc logic detected uncorrectable error while reading data from flash device.	RW	0x0

intr_en3

Enables corresponding interrupt bit in interrupt register for bank 3

Module Instance	Base Address	Register Address
i_nand_status	0xFFB80400	0xFFB80510

Offset: 0x110

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															erased_page RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
page_xfer_inc RW 0x0	pipe_cmd_err RW 0x0	rst_comp RW 0x1	int_act RW 0x0	unsup_cmd RW 0x0	locked_blk RW 0x0	pipe_cpybck_cmd_comp RW 0x0	erase_comp RW 0x0	program_comp RW 0x0	load_comp RW 0x0	erase_fail RW 0x0	program_fail RW 0x0	time_out RW 0x0	dma_cmd_comp RW 0x0	Reserved	ecc_uncor_err RW 0x0

intr_en3 Fields

Bit	Name	Description	Access	Reset
16	erased_page	If an erased page is detected on reads, this bit will be set. The detection of erased page is based on the number of 0's in the page. If the number of 0's in the page being read is less than the value in the erase_threshold (programmable register), an erased page is inferred and no un-correctable error will be flagged for that page. If ECC is disabled, the erased_page interrupt shall be set as explained above. If ECC is enabled, in addition to the above condition, only when the ECC logic detects an un-correctable error for that page will the erased_page interrupt be flagged. If the ECC logic detects a no-error or correctable error page, this erased page interrupt will not be set.	RW	0x0
15	page_xfer_inc	For every page of data transfer to or from the device, this bit will be set.	RW	0x0
14	pipe_cmd_err	A pipeline command sequence has been violated. This occurs when Map 01 page read/write address does not match the corresponding expected address from the pipeline commands issued earlier.	RW	0x0
13	rst_comp	A reset command has completed on this bank	RW	0x1
12	int_act	R/B pin of device transitioned from low to high	RW	0x0

Bit	Name	Description	Access	Reset
11	unsup_cmd	An unsupported command was received. This interrupt is set when an invalid command is received, or when a command sequence is broken.	RW	0x0
10	locked_blk	The address to program or erase operation is to a locked block and the operation failed due to this reason	RW	0x0
9	pipe_cpybck_cmd_comp	A pipeline command or a copyback bank command has completed on this particular bank	RW	0x0
8	erase_comp	Device erase operation complete	RW	0x0
7	program_comp	Device finished the last issued program command.	RW	0x0
6	load_comp	Device finished the last issued load command.	RW	0x0
5	erase_fail	Erase failure occurred in the device on issuance of a erase command. err_block_addr and err_page_addr contain the block address and page address that failed erase operation.	RW	0x0
4	program_fail	Program failure occurred in the device on issuance of a program command. err_block_addr and err_page_addr contain the block address and page address that failed program operation.	RW	0x0
3	time_out	Watchdog timer has triggered in the controller due to one of the reasons like device not responding or controller state machine did not get back to idle	RW	0x0
2	dma_cmd_comp	A data DMA command has completed on this bank	RW	0x0

Bit	Name	Description	Access	Reset
0	ecc_uncor_err	If set, Controller will interrupt processor when Ecc logic detects uncorrectable error.	RW	0x0

page_cnt3

Decrementing page count bank 3

Module Instance	Base Address	Register Address
i_nand_status	0xFFB80400	0xFFB80520

Offset: 0x120

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								value RO 0x0							

page_cnt3 Fields

Bit	Name	Description	Access	Reset
7:0	value	Maintains a decrementing count of the number of pages in the multi-page (pipeline and copyback) command being executed.	RO	0x0

err_page_addr3

Erred page address bank 3

Module Instance	Base Address	Register Address
i_nand_status	0xFFB80400	0xFFB80530

Offset: 0x130

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
value RO 0x0															

err_page_addr3 Fields

Bit	Name	Description	Access	Reset
15:0	value	Holds the page address that resulted in a failure on program or erase operation.	RO	0x0

err_block_addr3

Erred block address bank 3

Module Instance	Base Address	Register Address
i_nand_status	0xFFB80400	0xFFB80540

Offset: 0x140

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
value RO 0x0															

err_block_addr3 Fields

Bit	Name	Description	Access	Reset
15:0	value	Holds the block address that resulted in a failure on program or erase operation.	RO	0x0

nand_ecc Address Map

Module Instance	Base Address	End Address
i_nand_ecc	0xFFB80650	0xFFB806FF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
ecccorinfo_b01 on page 13-253	0x0	32	RO	0x0	ECC Error correction Information register. Controller updates this register when it completes a transaction. The values are held in this register till a new transaction completes.

Register	Offset	Width	Access	Reset Value	Description
ecccorinfo_b23 on page 13-255	0x10	32	RO	0x0	ECC Error correction Information register. Controller updates this register when it completes a transaction. The values are held in this register till a new transaction completes.

nand_ecc Summary

Base Address: 0xFFB80650

Register Address Offset	Bit Fields															
i_nand_ecc																
ecccorinfo_b01 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	uncor_err_b1 RO 0x0	max_errors_b1 RO 0x0							uncor_err_b0 RO 0x0	max_errors_b0 RO 0x0						
ecccorinfo_b23 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	uncor_err_b3 RO 0x0	max_errors_b3 RO 0x0							uncor_err_b2 RO 0x0	max_errors_b2 RO 0x0						

ecccorinfo_b01

ECC Error correction Information register. Controller updates this register when it completes a transaction. The values are held in this register

till a new transaction completes.

Module Instance	Base Address	Register Address
i_nand_ecc	0xFFB80650	0xFFB80650

Offset: 0x0

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
uncor_err_b1 RO 0x0	max_errors_b1 RO 0x0							uncor_err_b0 RO 0x0	max_errors_b0 RO 0x0						

ecccorinfo_b01 Fields

Bit	Name	Description	Access	Reset
15	uncor_err_b1	Uncorrectable error occurred while reading pages for last transaction in Bank1. Uncorrectable errors also generate interrupts in intr_statusx register.	RO	0x0
14:8	max_errors_b1	Maximum of number of errors corrected per sector in Bank1. This field is not valid for uncorrectable errors. A value of zero indicates that no ECC error occurred in last completed transaction.	RO	0x0

Bit	Name	Description	Access	Reset
7	uncor_err_b0	Uncorrectable error occurred while reading pages for last transaction in Bank0. Uncorrectable errors also generate interrupts in intr_statusx register.	RO	0x0
6:0	max_errors_b0	Maximum of number of errors corrected per sector in Bank0. This field is not valid for uncorrectable errors. A value of zero indicates that no ECC error occurred in last completed transaction.	RO	0x0

ecccorinfo_b23

ECC Error correction Information register. Controller updates this register when it completes a transaction. The values are held in this register till a new transaction completes.

Module Instance	Base Address	Register Address
i_nand_ecc	0xFFB80650	0xFFB80660

Offset: 0x10

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
uncor_err_b3 RO 0x0	max_errors_b3 RO 0x0							uncor_err_b2 RO 0x0	max_errors_b2 RO 0x0						

ecccorinfo_b23 Fields

Bit	Name	Description	Access	Reset
15	uncor_err_b3	Uncorrectable error occurred while reading pages for last transaction in Bank3. Uncorrectable errors also generate interrupts in intr_statusx register.	RO	0x0
14:8	max_errors_b3	Maximum of number of errors corrected per sector in Bank3. This field is not valid for uncorrectable errors. A value of zero indicates that no ECC error occurred in last completed transaction.	RO	0x0
7	uncor_err_b2	Uncorrectable error occurred while reading pages for last transaction in Bank2. Uncorrectable errors also generate interrupts in intr_statusx register.	RO	0x0
6:0	max_errors_b2	Maximum of number of errors corrected per sector in Bank2. This field is not valid for uncorrectable errors. A value of zero indicates that no ECC error occurred in last completed transaction.	RO	0x0

nand_dma Address Map

Module Instance	Base Address	End Address
i_nand_dma	0xFFB80700	0xFFB8FFFF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
dma_enable on page 13-261	0x0	32	RW	0x0	
dma_intr on page 13-262	0x20	32	RW	0x0	DMA interrupt register
dma_intr_en on page 13-263	0x30	32	RW	0x0	Enables corresponding interrupt bit in dma interrupt register
target_err_addr_lo on page 13-264	0x40	32	RO	0x0	Transaction address for which controller initiator interface received an ERROR target response.
target_err_addr_hi on page 13-265	0x50	32	RO	0x0	Transaction address for which controller initiator interface received an ERROR target response.
flash_burst_length on page 13-266	0x70	32	RW	0x1	
chip_interleave_enable_and_allow_int_reads on page 13-267	0x80	32	RW	0x110	
no_of_blocks_per_lun on page 13-269	0xA0	32	RW	0xF	
lun_status_cmd on page 13-271	0xB0	32	RW	0x7878	Indicates the command to be sent while checking status of the next LUN.
chnl_active on page 13-272	0x60	32	RO	0x0	Indicates CMD-DMA channel activity status
cmd_dma_channel_error on page 13-273	0xC0	32	RW	0x0	Bits indicating CMD-DMA channel receiving an error condition. To get more information on the error, s/w needs to read the status field of the descriptor.

Register	Offset	Width	Access	Reset Value	Description
cmd_dma_channel_error_en on page 13-274	0xD0	32	RW	0x0	Enable bits indicating CMD-DMA channel receiving an error condition. To get more information on the error, s/w needs to read the status field of the descriptor.
rescan_buffer_flag on page 13-275	0x90	32	RW	0x0	Rescan buffer flag.

nand_dma Summary

Base Address: 0xFFB80700

Register Address Offset	Bit Fields															
i_nand_dma																
dma_enable 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															flag RW 0x0
dma_intr 0x20	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved									cmdma_idle RW 0x0	Reserved	desc_comp_channel3 RW 0x0	desc_comp_channel2 RW 0x0	desc_comp_channel1 RW 0x0	desc_comp_channel0 RW 0x0	target_error RW 0x0

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
dma_intr_en 0x30	Reserved									cmdd ma_ idle RW 0x0	Rese rved	desc _ comp _ chan nel3 RW 0x0	desc _ comp _ chan nel2 RW 0x0	desc _ comp _ chan nel1 RW 0x0	desc _ comp _ chan nel0 RW 0x0	target_ error RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
target_err_ addr_lo 0x40	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	value RO 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
target_err_ addr_hi 0x50	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	value RO 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	polling_sync_counter_value RW 0x0															
flash_burst _length 0x70	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	polling_sync_counter_value RW 0x0								Reserved			cont inou s_ burs t RW 0x0	Reserved		value RW 0x1	
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
chip_interl eave_enable _and_allow_ int_reads 0x80	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							cmd_ dma_ erro r_ enab le RW 0x1	Reserved			allo w_ int_ read s_ with in_ luns RW 0x1	Reserved			chip_ interlea ve_ enable RW 0x0

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
no_of_blocks_per_lun 0xA0	Reserved			issue_read_before_sync RW 0x0	Reserved			update_sync_before_prog_comp RW 0x0	Reserved							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved												value RW 0xF			
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
lun_status_cmd 0xB0	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	value RW 0x7878															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
chnl_active 0x60	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved												channel3 RO 0x0	channel2 RO 0x0	channel1 RO 0x0	channel0 RO 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cmd_dma_channel_error 0xC0	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved												channel3 RW 0x0	channel2 RW 0x0	channel1 RW 0x0	channel0 RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cmd_dma_channel_error_en 0xD0	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved												channel3 RW 0x0	channel2 RW 0x0	channel1 RW 0x0	channel0 RW 0x0

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
rescan_buffer_flag 0x90	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved												flag RW 0x0			

dma_enable

Module Instance	Base Address	Register Address
i_nand_dma	0xFFB80700	0xFFB80700

Offset: 0x0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															flag RW 0x0

dma_enable Fields

Bit	Name	Description	Access	Reset
0	flag	Enables data DMA operation in the controller 1 - Enable DMA 0 - Disable DMA	RW	0x0

dma_intr

DMA interrupt register

Module Instance	Base Address	Register Address
i_nand_dma	0xFFB80700	0xFFB80720

Offset: 0x20

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									cmddma_idle RW 0x0	Reserved	desc_comp_channel3 RW 0x0	desc_comp_channel2 RW 0x0	desc_comp_channel1 RW 0x0	desc_comp_channel0 RW 0x0	target_error RW 0x0

dma_intr Fields

Bit	Name	Description	Access	Reset
6	cmddma_idle	Command DMA became IDLE after completing all descriptors	RW	0x0
4	desc_comp_channel3	Indicates CMD-DMA channel 3 descriptor execution done (updated when interrupt bit in cmd flags set) .	RW	0x0
3	desc_comp_channel2	Indicates CMD-DMA channel 2 descriptor execution done (updated when interrupt bit in cmd flags set) .	RW	0x0
2	desc_comp_channel1	Indicates CMD-DMA channel 1 descriptor execution done (updated when interrupt bit in cmd flags set) .	RW	0x0

Bit	Name	Description	Access	Reset
1	desc_comp_channel0	Indicates CMD-DMA channel 0 descriptor execution done (updated when interrupt bit in cmd flags set) .	RW	0x0
0	target_error	Controller initiator interface received an ERROR target response for a transaction.	RW	0x0

dma_intr_en

Enables corresponding interrupt bit in dma interrupt register

Module Instance	Base Address	Register Address
i_nand_dma	0xFFB80700	0xFFB80730

Offset: 0x30

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									cmddma_idle RW 0x0	Reserved	desc_comp_channel3 RW 0x0	desc_comp_channel2 RW 0x0	desc_comp_channel1 RW 0x0	desc_comp_channel0 RW 0x0	target_error RW 0x0

dma_intr_en Fields

Bit	Name	Description	Access	Reset
6	cmddma_idle	Interrupt processor when command DMA becomes IDLE after completing all descriptors.	RW	0x0

Bit	Name	Description	Access	Reset
4	desc_comp_channel3	Enable bit to indicates CMD-DMA channel 3 descriptor execution done (updated when interrupt bit in cmd flags set).	RW	0x0
3	desc_comp_channel2	Enable bit to indicates CMD-DMA channel 2 descriptor execution done (updated when interrupt bit in cmd flags set).	RW	0x0
2	desc_comp_channel1	Enable bit to indicates CMD-DMA channel 1 descriptor execution done (updated when interrupt bit in cmd flags set).	RW	0x0
1	desc_comp_channel0	Enable bit to indicates CMD-DMA channel 0 descriptor execution done (updated when interrupt bit in cmd flags set).	RW	0x0
0	target_error	Controller initiator interface received an ERROR target response for a transaction.	RW	0x0

target_err_addr_lo

Transaction address for which controller initiator interface received an ERROR target response.

Module Instance	Base Address	Register Address
i_nand_dma	0xFFB80700	0xFFB80740

Offset: 0x40

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
value RO 0x0															

target_err_addr_lo Fields

Bit	Name	Description	Access	Reset
15:0	value	Least significant 16 bits	RO	0x0

target_err_addr_hi

Transaction address for which controller initiator interface received an ERROR target response.

Module Instance	Base Address	Register Address
i_nand_dma	0xFFB80700	0xFFB80750

Offset: 0x50

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
value RO 0x0															

target_err_addr_hi Fields

Bit	Name	Description	Access	Reset
15:0	value	Most significant 16 bits	RO	0x0

flash_burst_length

Module Instance	Base Address	Register Address
i_nand_dma	0xFFB80700	0xFFB80770

Offset: 0x70

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
polling_sync_counter_value RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
polling_sync_counter_value RW 0x0								Reserved			continous_burst RW 0x0	Reserved		value RW 0x1	

flash_burst_length Fields

Bit	Name	Description	Access	Reset
31:8	polling_sync_counter_value	Number of cycles CMDDMA channel has to wait before polling the SYNC Pointer again. If this counter value is 0, no polling is done.	RW	0x0
4	continous_burst	When this bit is set, the Data DMA will burst the entire page from/to the flash device. Please make sure that the host system can provide/sink data at a fast pace to avoid unnecessary pausing of data on the device interface.	RW	0x0

Bit	Name	Description	Access	Reset
1:0	value	Sets the burst used by data dma for transferring data to/from flash device. This burst length is different and is larger than the burst length on the host bus so that larger amount of data can be transferred to/from device, descreasing controller data transfer overhead in the process. 00 - 64 bytes, 01 - 128 bytes, 10 - 256 bytes, 11 - 512 bytes. The host burst size multiplied by the number of outstanding requests on the host side should be greater than equal to this value. If not, the device side burst length will be equal to host side burst length.	RW	0x1

chip_interleave_enable_and_allow_int_reads

Module Instance	Base Address	Register Address
i_nand_dma	0xFFB80700	0xFFB80780

Offset: 0x80

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							cmd_dma_error_enable RW 0x1	Reserved			allow_int_reads_within_luns RW 0x1	Reserved			chip_interleave_enable RW 0x0

chip_interleave_enable_and_allow_int_reads Fields

Bit	Name	Description	Access	Reset
8	cmd_dma_error_enable	This bit informs the CDMA channels to stop working on any new MAP10 Command DMAcommands from the host after encountering an error situation till the error bit for that corresponding channel is cleared in the cmd_dma_channel_error register by f/w. When the CDMA channel encounters an error, it will set the corresponding error bit in cmd_dma_channel_error register If this bit is set, the channel will stop executing any further commands till f/w comes and clears the error bit in the cmd_dma_channel_error_register. If this bit is not set, controller will still keep on executing new commands issued from f/w.	RW	0x1

Bit	Name	Description	Access	Reset
4	allow_int_reads_ within_luns	This bit informs the controller to enable or disable simultaneous read accesses to different LUNS in the same bank. This bit is of importance only if the controller supports interleaved operations among LUNs and if the device has multiple LUNS. If the bit is disabled, the controller will send read commands to different LUNS of of the same bank only sequentially and if enabled, the controller will issue simulta-neous read accesses to LUNS of same bank if required. [list][*]1 - Enable [*]0 - Disable[/list]	RW	0x1
0	chip_interleave_ enable	This bit informs the controller to enable or disable interleaving among banks/LUNS to increase the net performance of the controller. [list][*]1 - Enable interleaving [*]0 - Disable Interleaving[/list]	RW	0x0

no_of_blocks_per_lun

Module Instance	Base Address	Register Address
i_nand_dma	0xFFB80700	0xFFB807A0

Offset: 0xA0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved			issue_read_before_sync RW 0x0	Reserved			update_sync_before_prog_comp RW 0x0	Reserved							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												value RW 0xF			

no_of_blocks_per_lun Fields

Bit	Name	Description	Access	Reset
28	issue_read_before_sync	Issue LOAD cmd to flash core even if SYNC condition is not satisfied. But the data is read from the device (for this load) only after the SYNC condition has been satisfied. If this value is 0, CMD DMA waits for SYNC before issuing a READ command. This bit should be set to 0 if the controller is being accessed in non-Command DMA mode.	RW	0x0
24	update_sync_before_prog_comp	Update SYNC Pointer after the data is written to flash and dont wait for program to complete. If this value is 0, CMD DMA waits for page program to get over before updating the sync pointer. This bit should be set to 0 if the controller is being accessed in non-Command DMA mode.	RW	0x0

Bit	Name	Description	Access	Reset
3:0	value	Indicates the first block of next LUN. This information is used for extracting the target LUN during LUN interleaving. After Initialization, if the controller detects an ONFi device, this field is automatically updated by the controller. For other devices, software will need to write to this register for proper interleaving. The value in this register is interpreted as follows- [list][*]0 - Next LUN starts from 1024. [*]1 - Next LUN starts from 2048. [*]2 - Next LUN starts from 4096 and so on... [/list]	RW	0xF

lun_status_cmd

Indicates the command to be sent while checking status of the next LUN.

Module Instance	Base Address	Register Address
i_nand_dma	0xFFB80700	0xFFB807B0

Offset: 0xB0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
value RW 0x7878															

lun_status_cmd Fields

Bit	Name	Description	Access	Reset
15:0	value	[list][*]7:0 - Indicates the command to check the status of the first LUN/Die. [*]15:8 - Indicates the command to check the status of the other LUN/Die.[/list]	RW	0x7878

chnl_active

Indicates CMD-DMA channel activity status

Module Instance	Base Address	Register Address
i_nand_dma	0xFFB80700	0xFFB80760

Offset: 0x60

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												channel3 RO 0x0	channel2 RO 0x0	channel1 RO 0x0	channel0 RO 0x0

chnl_active Fields

Bit	Name	Description	Access	Reset
3	channel3	CMD-DMA channel 3 is active	RO	0x0
2	channel2	CMD-DMA channel 2 is active	RO	0x0
1	channel1	CMD-DMA channel 1 is active	RO	0x0
0	channel0	CMD-DMA channel 0 is active	RO	0x0

cmd_dma_channel_error

Bits indicating CMD-DMA channel receiving an error condition. To get more information on the error, s/w needs to read the status field of the descriptor.

Module Instance	Base Address	Register Address
i_nand_dma	0xFFB80700	0xFFB807C0

Offset: 0xC0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												channel3 RW 0x0	channel2 RW 0x0	channel1 RW 0x0	channel0 RW 0x0

cmd_dma_channel_error Fields

Bit	Name	Description	Access	Reset
3	channel3	CMD-DMA channel 3 received an error.	RW	0x0
2	channel2	CMD-DMA channel 2 received an error.	RW	0x0
1	channel1	CMD-DMA channel 1 received an error.	RW	0x0
0	channel0	CMD-DMA channel 0 received an error.	RW	0x0

cmd_dma_channel_error_en

Enable bits indicating CMD-DMA channel receiving an error condition. To get more information on the error, s/w needs to read the status field of the descriptor.

Module Instance	Base Address	Register Address
i_nand_dma	0xFFB80700	0xFFB807D0

Offset: 0xD0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												channel3 RW 0x0	channel2 RW 0x0	channel1 RW 0x0	channel0 RW 0x0

cmd_dma_channel_error_en Fields

Bit	Name	Description	Access	Reset
3	channel3	enable bit for CMD-DMA channel 3 receiving an error	RW	0x0
2	channel2	enable bit for CMD-DMA channel 2 receiving an error	RW	0x0
1	channel1	enable bit for CMD-DMA channel 1 receiving an error	RW	0x0
0	channel0	enable bit for CMD-DMA channel 0 receiving an error	RW	0x0

rescan_buffer_flag

Rescan buffer flag.

Module Instance	Base Address	Register Address
i_nand_dma	0xFFB80700	0xFFB80790

Offset: 0x90

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												flag RW 0x0			

rescan_buffer_flag Fields

Bit	Name	Description	Access	Reset
3:0	flag	This register can be used to force rescan of buffer flags in any of the cmd-dma channels. The bit index decides the Channel number. Cmd-dma would rescan the buffer flag for matching condition and it executes the descriptor if it is ready. Hardware clears this register after generating the trigger event.	RW	0x0

nand_NANDDATA Address Map

This address space is allocated for indexed addressing by the NAND flash controller. For more information, please refer to the NAND Flash Controller chapter of the Arria 10 Hard Processor System Technical Reference Manual.

Module Instance	Base Address	End Address
i_nand_NANDDATA	0xFFB90000	0xFFC01FFF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Document Revision History

Table 13-18: Document Revision History

Date	Version	Changes
October 2016	2016.10.28	Maintenance release
May 2016	2016.05.27	Added a link to the *Supported Flash Devices for Arria 10 SoC* webpage.
May 2016	2016.05.03	Added information about determining how many CE/RB signals are available based on the selected pins.
November 2015	2015.11.02	• Updated the Interrupt and DMA Enabling section to recommend reading back a register to ensure clearing an interrupt status. • Removed the reference to a missing figure from the `cs_setup_cnt` description. • Documented the behavior of the `wp_n` bit for when it will or will not be available.
May 2015	2015.05.04	Added information about clearing out the ECC before the feature is enabled
December 2014	2014.12.15	Maintenance release
August 2014	2014.08.18	Initial release

SD/MMC Controller 14

2016.10.28

a10_5v4

The hard processor system (HPS) provides a Secure Digital/Multimedia Card (SD/MMC) controller for interfacing to external SD and MMC flash cards, secure digital I/O (SDIO) devices, and Consumer Electronics Advanced Transport Architecture (CE-ATA) hard drives. The SD/MMC controller enables you to store boot images and boot the processor system from the removable flash card. You can also use the flash card to expand the on-board storage capacity for larger applications or user data. Other applications include interfacing to embedded SD (eSD) and embedded MMC (eMMC) non-removable flash devices.

The SD/MMC controller is based on the Synopsys DesignWare Mobile Storage Host (SD/MMC controller) controller.[32]

This document refers to SD/SDIO commands, which are documented in detail in the *Physical Layer Simplified Specification, Version 3.01* and the *SDIO Simplified Specification, Version 2.00* described on the SD Association website.

Related Information

SD Association

To learn more about how SD technology works, visit the SD Association website (www.sdcard.org).

Features of the SD/MMC Controller

The HPS SD/MMC controller offers the following features:

[32] Portions © 2016 Synopsys, Inc. Used with permission. All rights reserved. Synopsys & DesignWare are registered trademarks of Synopsys, Inc. All documentation is provided "as is" and without any warranty. Synopsys expressly disclaims any and all warranties, express, implied, or otherwise, including the implied warranties of merchantability, fitness for a particular purpose, and non-infringement, and any warranties arising out of a course of dealing or usage of trade.

†Paragraphs marked with the dagger (†) symbol are Synopsys Proprietary. Used with permission.

- Supports HPS boot from mobile storage
- Supports the following standards or card types:[33]
  - SD, including eSD—version 3.0
  - SDIO, including embedded SDIO (eSDIO)—version 3.0
  - CE-ATA—version 1.1
- Supports various types of multimedia cards (MMC version 4.41 and eMMC version 4.5[34])[35]
  - MMC: 1-bit data bus
  - Reduced-size MMC (RSMMC): 1-bit and 4-bit data bus
  - MMCPlus: 1-bit, 4-bit, and 8-bit data bus
  - MMCMobile: 1-bit data bus
  - Embedded MMC (eMMC) version 4.5: 1-bit, 4-bit, and 8-bit data bus
- Supports the Micron MTFC16GJDDQ-4M IT (16 GB) eMMC flash memory that is verified to work with the HPS.
- Supports the Kingston® 4 GB SD Micro flash card
- Integrated descriptor-based direct memory access (DMA)
- Internal 4 KB receive and transmit FIFO buffer

Unsupported Features

- Card Detect is only supported on interfaces routed via the FPGA fabric. The Card Detect Interface signals are not included if the interface is pinned out to HPS I/O.
- The SD/MMC controller does not directly support voltage switching, card interrupts, or back-end power control of eSDIO card devices. However, you can connect these signals to general-purpose I/Os (GPIOs).
- The SD/MMC controller does not contain a reset output as part of the external card interface. To reset the flash card device, consider using a general purpose output pin.

Related Information

- **MMC Support Matrix** on page 14-3
  For more information on what is supported, refer to the MMC Support Matrix table.
- **Supported Flash Devices for Arria 10 SoC**
  For more information, refer to the supported SD/SDHC/SDXC/MMC/eMMC flash devices section on this page.

(33) SD and SDIO do not support SDR50, SDR104, and DDR50 modes.
(34) This is a new bus mode for Arria 10.
(35) The MMC and RSMMC does not support DDR. However, the MMCPlus, MMCMobile, and eMMC do support DDR, but not by the HPS.

SD Card Support Matrix

Table 14-1: SD Card Support Matrix

Device Card Type	Voltages Supported[36]		Bus Modes Supported			Bus Speed Modes Supported			
						Default Speed	High Speed	SDR12	SDR25[37]
	3.0 V	1.8 V[38]	1 bit	4 bit	8 bit	12.5 MBps 25 MHz	25 MBps 50 MHz	12.5 MBps 25 MHz	25 MBps 50 MHz
SDSC (SD)	√		√			√	√		
SDHC	√	√	√	√		√	√	√	√
SDXC	√	√	√	√		√	√	√	√
eSD	√	√	√	√		√	√	√	√
SDIO	√	√	√	√		√	√	√	√
eSDIO	√	√	√	√	√	√	√	√	√

Note: Card form factors (such as mini and micro) are not enumerated in the above table because they do not impact the card interface functionality.

MMC Support Matrix

Table 14-2: MMC Support Matrix

Card Device Type	Max Clock Speed (MHz)	Max Data Rate (MBps)	Voltages Supported		Bus Modes Supported			Bus Speed Modes Supported	
			3.3 V	1.8 V	1 bit	4 bit	8 bit	Default Speed	High Speed
MMC	20	2.5	√		√			√	
RSMMC	20	10	√		√	√		√	√
MMCPlus	50[39]	25	√		√	√	√	√	√
MMCMobile	50	6.5	√	√	√			√	√
eMMC	50	25	√	√	√	√	√	√	√

[36] Because SD cards initially operate at 3.0 V and can switch to 1.8V after power-up and the BSEL values are constant during the boot process, transceivers are required to support level-shifting and isolation.

[37] SDR25 speed mode requires 1.8-V signaling. Note that even if a card supports UHS-I modes (for example SDR50, SDR104, DDR50) it can still communicate at the lower speeds (for example SDR12, SDR25).

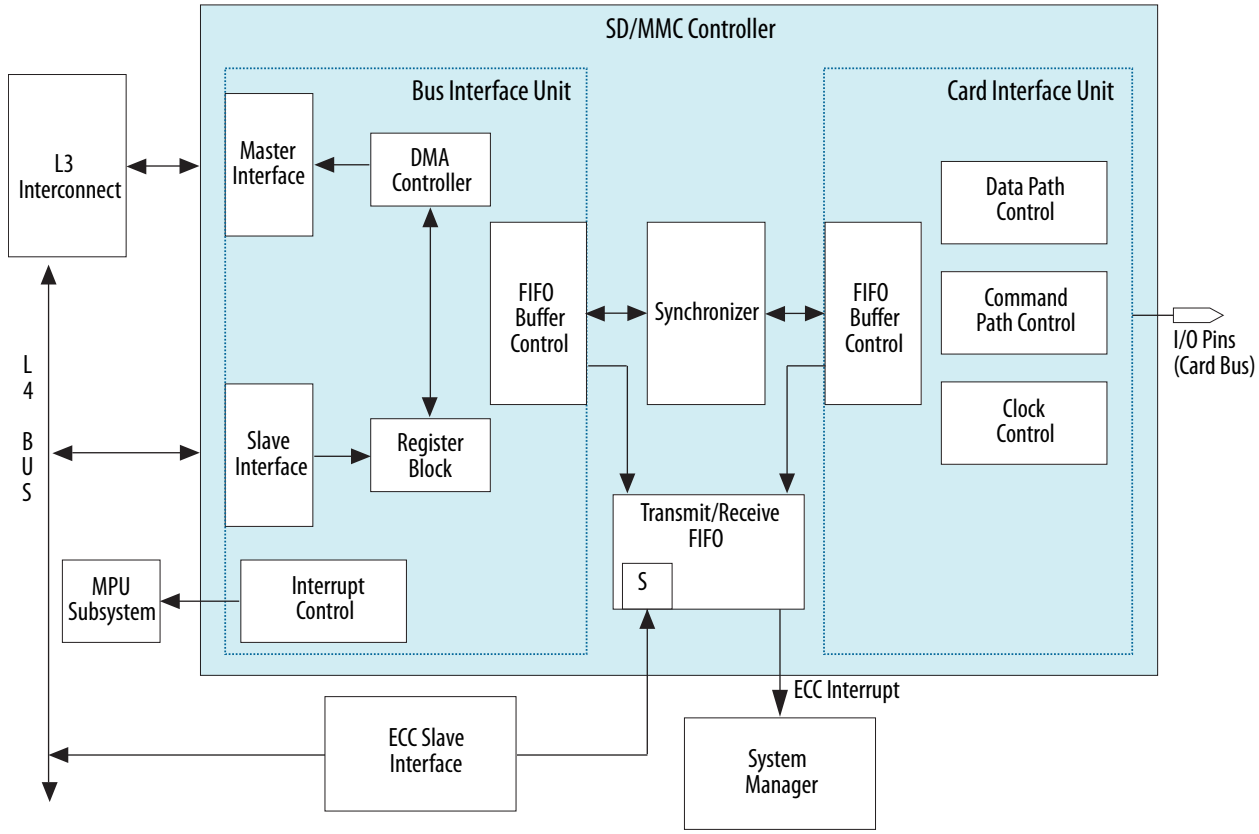
[38] Where supported, external transceivers are needed to switch the voltage.

[39] Supports a maximum clock rate of 50 MHz instead of 52 MHz (specified in MMC specification).

SD/MMC Controller Block Diagram and System Integration

The SD/MMC controller includes a bus interface unit (BIU) and a card interface unit (CIU). The BIU provides a slave interface for a host to access the control and status registers (CSRs). Additionally, this unit also provides independent FIFO buffer access through a DMA interface. The DMA controller is responsible for exchanging data between the system memory and FIFO buffer. The DMA registers are accessible by the host to control the DMA operation. The CIU supports the SD, MMC, and CE-ATA protocols on the controller, and provides clock management through the clock control block. The interrupt control block for generating an interrupt connects to the generic interrupt controller in the ARM Cortex-A9 microprocessor unit (MPU) subsystem.

Figure 14-1: SD/MMC Controller Connectivity



SD/MMC Controller Signal Description

The following table shows the SD/MMC controller signals that are routed to the FPGA and the HPS I/O.

Note: The last five signals in the table are not routed to HPS I/O, but only to the FPGA. For more information on how each of the signals are routed to the FPGA and HPS I/O pins, refer to the *HPS Component Interfaces* chapter.

Table 14-3: SD/MMC Controller Interface I/O Pins

Signal	Width	Direction	Description
sdmmc_cclk_out	1	Out	Clock from controller to the card
sdmmc_cmd_i	1	In	Card command
sdmmc_cmd_o	1	Out	
sdmmc_cmd_oe		Out	
sdmmc_pwr_ena_o	1	Out	External device power enable
sdmmc_data_i	8	In	Card data
sdmmc_data_o	8	Out	
sdmmc_data_oe	1	Out	
sdmmc_cdn_i	1	In	Card detect signal
sdmmc_wp_i	1	In	Card write protect signal
sdmmc_vs_o	1	Out	Voltage switching between 3.3V and 1.8V
sdmmc_rstn_o	1	Out	Card reset signal used in MMC mode
sdmmc_card_intn_i	1	In	Card interrupt signal

Related Information

HPS Component Interfaces on page 28-8
For more information which SD/MMC Controller signals are routed to the FPGA and the HPS I/O, refer to this chapter.

Functional Description of the SD/MMC Controller

This section describes the SD/MMC controller components and how the controller operates.

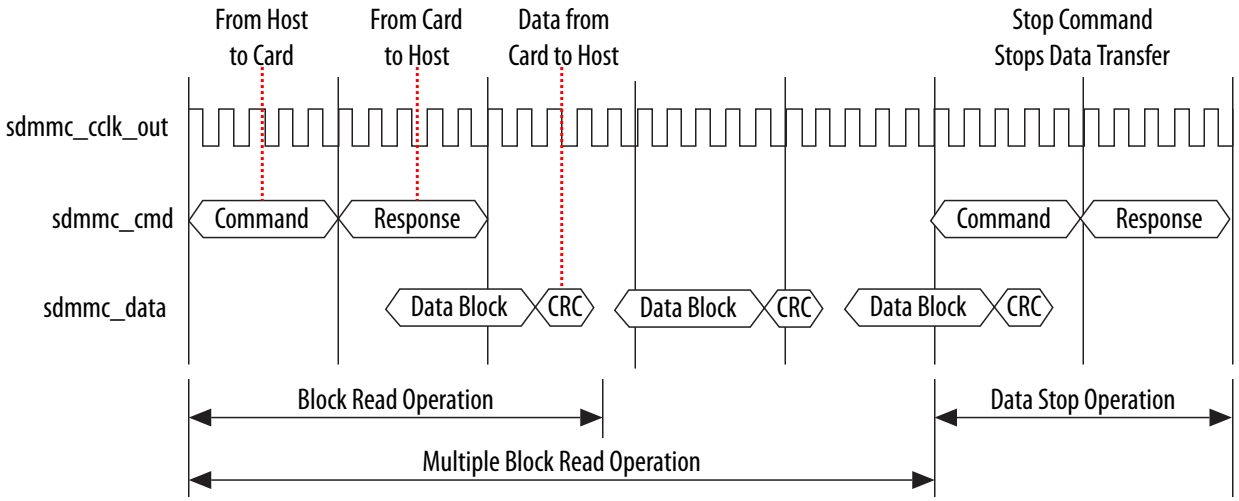
SD/MMC/CE-ATA Protocol

The SD/MMC/CE-ATA protocol is based on command and data bit streams that are initiated by a start bit and terminated by a stop bit. Additionally, the SD/MMC controller provides a reference clock and is the only master interface that can initiate a transaction.†

- Command—a token transmitted serially on the CMD pin that starts an operation.†
- Response—a token from the card transmitted serially on the CMD pin in response to certain commands.†
- Data—transferred serially using the data pins for data movement commands.†

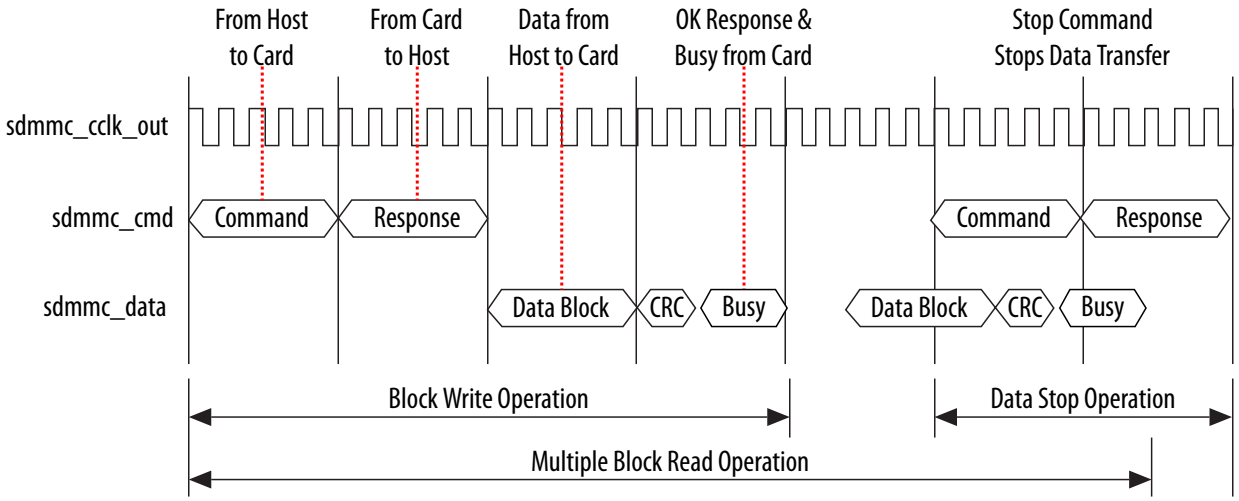
In the following figure, the clock is a representative only and does not show the exact number of clock cycles.

Figure 14-2: Multiple–Block Read Operation†



The following figure illustrates an example of a command token sent by the host in a multiple-block write operation.

Figure 14-3: Multiple-–Block Write Operation†



BIU

The Bus Interface Unit (BIU) interfaces with the Card Interface Unit (CIU), and is connected to the level 3 (L3) interconnect and level 4 (L4) peripheral buses. The BIU consists of the following primary functional blocks, which are defined in the following sections:

- Slave interface
- Register block
- FIFO buffer
- Interrupt control
- Internal DMA controller

Slave Interface

The host processor accesses the SD/MMC controller registers and data FIFO buffers through the slave interface.

Register Block

The register block is part of the BIU and provides read and write access to the CSRs.

All registers reside in the BIU clock domain, `l4_mp_clk`. When a command is sent to a card by setting the start command bit (`start_cmd`) of the command register (`cmd`) to 1, all relevant registers needed for the CIU operation are copied to the CIU block. During this time, software must not write to the registers that are transferred from the BIU to the CIU. The software must wait for the hardware to reset the `start_cmd` bit to 0 before writing to these registers again. The register unit has a hardware locking feature to prevent illegal writes to registers.

Registers Locked Out Pending Command Acceptance

After a command start is issued by setting the `start_cmd` bit of the `cmd` register, the following registers cannot be rewritten until the command is accepted by the CIU:†

- Command (`cmd`)†
- Command argument (`cmdarg`)†
- Byte count (`bytcnt`)†
- Block size (`blksiz`)†
- Clock divider (`clkdiv`)†
- Clock enable (`clkena`)†
- Clock source (`clksrc`)†
- Timeout (`tmout`)†
- Card type (`ctype`)†

The hardware resets the `start_cmd` bit after the CIU accepts the command. If a host write to any of these registers is attempted during this locked time, the write is ignored and the hardware lock write error bit (`hle`) is set to 1 in the raw interrupt status register (`rintsts`). Additionally, if the interrupt is enabled and not masked for a hardware lock error, an interrupt is sent to the host.†

Once a command is accepted, you can send another command to the CIU—which has a one-deep command queue—under the following conditions:†

- If the previous command is not a data transfer command, the new command is sent to the SD/MMC/CE-ATA card once the previous command completes.†
- If the previous command is a data transfer command and if the wait previous data complete bit (`wait_prvdata_complete`) of the `cmd` register is set to 1 for the new command, the new command is sent to the SD/MMC/CE-ATA card only when the data transfer completes.†
- If the `wait_prvdata_complete` bit is 0, the new command is sent to the SD/MMC/CE-ATA card as soon as the previous command is sent. Typically, use this feature to stop or abort a previous data transfer or query the card status in the middle of a data transfer.†

Interrupt Controller Unit

The interrupt controller unit generates an interrupt that depends on the `rintsts` register, the interrupt mask register (`intmask`), and the interrupt enable bit (`int_enable`) of the control register (`ctrl`). Once an interrupt condition is detected, the controller sets the corresponding interrupt bit in the `rintsts` register. The bit in the `rintsts` register remains set until the software clears the bit by writing a 1 to the interrupt bit; writing a 0 leaves the bit untouched.

The interrupt controller unit generates active high, level sensitive interrupts that are asserted only when at least one bit in the `rintsts` register is set to 1, the corresponding `intmask` register bit is 1, and the `int_enable` bit of the `ctrl` register is 1.

The `int_enable` bit of the `ctrl` register is cleared during a power-on reset, and the `intmask` register bits are set to 0x0000000, which masks all the interrupts.

Table 14-4: Interrupt Status Register Bits†

Bits	Interrupt	Description
16	SDIO Interrupts†	Interrupts from SDIO cards.†
15	End Bit Error (read)/Write no CRC (EBE)†	Error in end-bit during read operation, or no data CRC received during write operation.† **Note:** For MMC CMD19, there may be no CRC status returned by the card. Hence, EBE is set for CMD19. The application should not treat this as an error. †
14	Auto Command Done (ACD)†	Stop/abort commands automatically sent by card unit and not initiated by host; similar to Command Done (CD) interrupt. † **Recommendation:** Software typically need not enable this for non CE-ATA accesses; Data Transfer Over (DTO) interrupt that comes after this interrupt determines whether data transfer has correctly competed. For CE-ATA accesses, if the software sets `send_auto_stop_ccsd` bit in the control register, then software should enable this bit.†
13	Start Bit Error (SBE)	Error in data start bit when data is read from a card. In 4-bit mode, if all data bits do not have start bit, then this error is set.
12	Hardware Locked write Error (HLE)†	During hardware-lock period, write attempted to one of locked registers. †

Bits	Interrupt	Description
11	FIFO Underrun/Overrun Error (FRUN)†	Host tried to push data when FIFO was full, or host tried to read data when FIFO was empty. Typically this should not happen, except due to error in software. † Card unit never pushes data into FIFO when FIFO is full, and pop data when FIFO is empty. † If IDMAC (Internal Direct Memory Access Controller) is enabled, FIFO underrun/overrun can occur due to a programming error on MSIZE and watermark values in FIFOTH register; for more information, refer to *Internal Direct Memory Access Controller (IDMAC)* section in the "Synopsys DesignWare Cores Mobile Storage Host Databook".†
10	Data Starvation by Host Timeout (HTO)†	To avoid data loss, card clock out (`cclk_out`) is stopped if FIFO is empty when writing to card, or FIFO is full when reading from card. Whenever card clock is stopped to avoid data loss, data-starvation timeout counter is started with data-timeout value. This interrupt is set if host does not fill data into FIFO during write to card, or does not read from FIFO during read from card before timeout period. † Even after timeout, card clock stays in stopped state, with CIU state machines waiting. It is responsibility of host to push or pop data into FIFO upon interrupt, which automatically restarts `cclk_out` and card state machines. † Even if host wants to send stop/abort command, it still must ensure to push or pop FIFO so that clock starts in order for stop/abort command to send on cmd signal along with data that is sent or received on data line. †
9	Data Read Timeout (DRTO)/Boot Data Start (BDS)†	• In Normal functioning mode: Data read timeout (DRTO) Data timeout occurred. Data Transfer Over (DTO) also set if data timeout occurs. † • In Boot Mode: Boot Data Start (BDS) When set, indicates that SD/MMC controller has started to receive boot data from the card. A write to this register with a value of 1 clears this interrupt.†

Bits	Interrupt	Description
8	Response Timeout (RTO)/ Boot Ack Received (BAR)†	• In Normal functioning mode: Response timeout (RTO) Response timeout occurred. Command Done (CD) also set if response timeout occurs. If command involves data transfer and when response times out, no data transfer is attempted by SD/MMC controller.† • In Boot Mode: Boot Ack Received (BAR) When expect_boot_ack is set, on reception of a boot acknowledge pattern—0-1-0—this interrupt is asserted. A write to this register with a value of 1 clears this interrupt.†
7	Data CRC Error (DCRC)†	Received Data CRC does not match with locally-generated CRC in CIU; expected when a negative CRC is received. †
6	Response CRC Error (RCRC)†	Response CRC does not match with locally-generated CRC in CIU.†
5	Receive FIFO Data Request (RXDR)†	Interrupt set during read operation from card when FIFO level is greater than Receive-Threshold level.† **Recommendation:** In DMA modes, this interrupt should not be enabled.† ISR, in non-DMA mode: `pop RX_WMark + 1 data from FIFO` †

4	Transmit FIFO Data Request (TXDR)†	Interrupt set during write operation to card when FIFO level reaches less than or equal to Transmit-Threshold level.†
		Recommendation: In DMA modes, this interrupt should not be enabled.†
		ISR in non-DMA mode: †
		`if (pending_bytes > \†`
		`(FIFO_DEPTH - TX_WMark))†`
		`push (FIFO_DEPTH - \†`
		`TX_WMark) data into FIFO†`
		`else†`
		`push pending_bytes data \†`
		`into FIFO†`

Bits	Interrupt	Description
3	Data Transfer (DTO)†	Data transfer completed, even if there is Start Bit Error or CRC error. This bit is also set when "read data-timeout" occurs or CCS is sampled from CE-ATA device.† **Recommendation:** In non-DMA mode, when data is read from card, on seeing interrupt, host should read any pending data from FIFO. In DMA mode, DMA controllers guarantee FIFO is flushed before interrupt.† **Note:** DTO bit is set at the end of the last data block, even if the device asserts MMC busy after the last data block.†
2	Command Done (CD)†	Command sent to card and received response from card, even if Response Error or CRC error occurs. Also set when response timeout occurs or CCSD sent to CE-ATA device.†
1	Response Error (RE)†	Error in received response set if one of following occurs:† • Transmission bit != 0† • Command index mismatch† • End-bit != 1†
0	Card-Detect (CDT)†	When one or more cards inserted or removed, this interrupt occurs. Software should read card-detect register (CDETECT, 0x50) to determine current card status.† **Recommendation:** After power-on and before enabling interrupts, software should read card detect register and store it in memory. When interrupt occurs, it should read card detect register and compare it with value stored in memory to determine which card(s) were removed/inserted. Before exiting ISR, software should update memory with new card-detect value.†

Interrupt Setting and Clearing

The SDIO Interrupts, Receive FIFO Data Request, and Transmit FIFO Data Request interrupts are set by level-sensitive interrupt sources. Therefore, the interrupt source must be first cleared before you can reset the interrupt's corresponding bit in the `rintsts` register to 0.†

For example, on receiving the Receive FIFO Data Request interrupt, the FIFO buffer must be emptied so that the FIFO buffer count is not greater than the RX watermark, which causes the interrupt to be triggered.†

The rest of the interrupts are triggered by single clock-pulse-width sources.†

FIFO Buffer

The SD/MMC controller has a 4 KB data FIFO buffer for storing transmit and receive data.

The FIFO has an ECC controller built-in to provide ECC protection. The ECC controller is able to detect single-bit and double-bit errors, and correct the single-bit errors. The ECC operation and functionality is programmable through the ECC register slave interface. The ECC register slave interface provides host access to configure the ECC logic as well as inject bit errors into the memory. It also provides the host access to memory initialization hardware used to clear out the memory contents including the ECC bits. The ECC controller generates interrupts upon occurrences of single- and double-bit errors, and the interrupt signals are connected to the system manager.

Note: Since SD/MMC has multiple memories, it must initialize both memories explicitly. Initialization is controlled by software through the ECC Control (CTRL) register. This process cannot be interrupted or stopped once it starts; hence software must wait for the initialization complete bit to be set in the Initialization Status (`INITSTAT`) register. Memory accesses are allowed after the initialization process is complete.

Related Information

- **System Manager** on page 5-1
- **Error Checking and Correction Controller** on page 11-1
  For more information about ECC, refer to the *Error Checking and Correction Controller* chapter of the Arria 10 Device Handbook.
- **Memory Data Initialization** on page 11-5
  For more information about clearing memory data before enabling ECC, refer to "Memory Data Initialization" in the Error Checking and Correction section.

Internal DMA Controller

Internal DMA controller (AHB Master) enables the core to act as a Master on the AHB to transfer data to and from the AHB.

- Supports 32-bit data
- Supports split, retry, and error AHB responses, but does not support wrap
- Configurable for little-endian or big-endian mode
- Allows the selection of AHB burst type through software

The internal DMA controller has a CSR and a single transmit or receive engine, which transfers data from system memory to the card and vice versa. The controller uses a descriptor mechanism to efficiently move data from source to destination with minimal host processor intervention. You can configure the controller to interrupt the host processor in situations such as transmit and receive data transfer completion from the card, as well as other normal or error conditions. The DMA controller and the host driver communicate through a single data structure.†

The internal DMA controller transfers the data received from the card to the data buffer in the system memory, and transfers transmit data from the data buffer in the memory to the controller's FIFO buffer. Descriptors that reside in the system memory act as pointers to these buffers.†

A data buffer resides in the physical memory space of the system memory and consists of complete or partial data. The buffer status is maintained in the descriptor. Data chaining refers to data that spans multiple data buffers. However, a single descriptor cannot span multiple data buffers.†

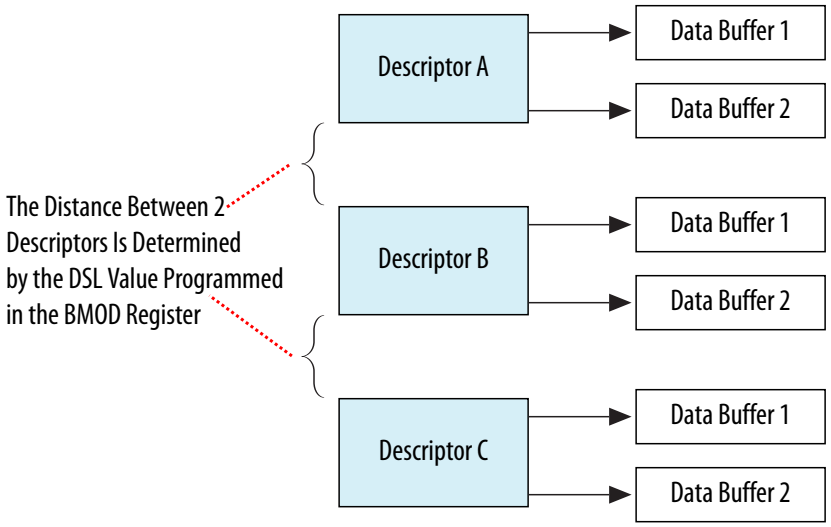
A single descriptor is used for both reception and transmission. The base address of the list is written into the descriptor list base address register (`dbaddr`). A descriptor list is forward linked. The last descriptor can point back to the first entry to create a ring structure. The descriptor list resides in the physical memory address space of the host. Each descriptor can point to a maximum of two data buffers.†

Internal DMA Controller Descriptors

The internal DMA controller uses these types of descriptor structures:†

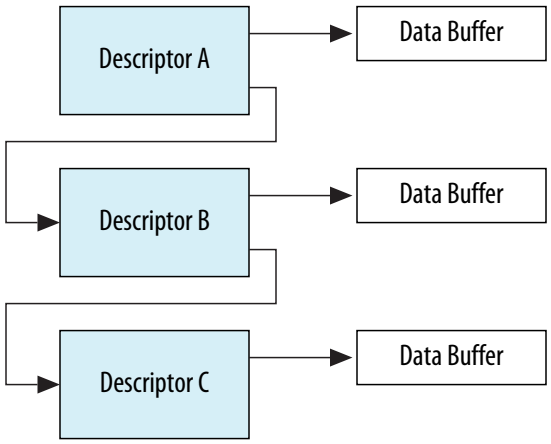
- Dual-buffer structure—The distance between two descriptors is determined by the skip length value written to the descriptor skip length field (`dsl`) of the bus mode register (`bmod`).†

Figure 14-4: Dual-Buffer Descriptor Structure†



- Chain structure—Each descriptor points to a unique buffer, and to the next descriptor in a linked list.†

Figure 14-5: Chain Descriptor Structure†



Internal DMA Controller Descriptor Address

The descriptor address must be aligned to the 32-bit bus. Each descriptor contains 16 bytes of control and status information.†

Table 14-5: Descriptor Format

Name	Off-set	31	30	29:27	26	25:14	13	12:7	6	5	4	3	2	1	0
DES0	0	OWN	CES	—						ER	CH	FS	LD	DIC	—
DES1	4	—				BS2		BS1							
DES2	8	BAP1													
DES3	12	BAP2 or Next Descriptor Address													

Related Information

Internal DMA Controller Descriptor Fields on page 14-14
Refer to this table for information about each of the bits of the descriptor.

Internal DMA Controller Descriptor Fields

The DES0 field in the internal DMA controller descriptor contains control and status information.

Table 14-6: Internal DMA Controller DES0 Descriptor Field†

Bits	Name	Description
31	OWN	When set to 1, this bit indicates that the descriptor is owned by the internal DMA controller. When this bit is set to 0, it indicates that the descriptor is owned by the host. The internal DMA controller resets this bit to 0 when it completes the data transfer.
30	Card Error Summary (CES)	The CES bit indicates whether a transaction error occurred. The CES bit is the logical OR of the following error bits in the `rintsts` register. • End-bit error (`ebe`) • Response timeout (`rto`) • Response CRC (`rcrc`) • Start-bit error (`sbe`) • Data read timeout (`drto`) • Data CRC for receive (`dcrc`) • Response error (`re`)
29:6	Reserved	—

Bits	Name	Description
5	End of Ring (ER)	When set to 1, this bit indicates that the descriptor list reached its final descriptor. The internal DMA controller returns to the base address of the list, creating a descriptor ring. ER is meaningful for only a dual-buffer descriptor structure.
4	Second Address Chained (CH)	When set to 1, this bit indicates that the second address in the descriptor is the next descriptor address rather than the second buffer address. When this bit is set to 1, BS2 (DES1[25:13]) must be all zeros.
3	First Descriptor (FD)	When set to 1, this bit indicates that this descriptor contains the first buffer of the data. If the size of the first buffer is 0, next descriptor contains the beginning of the data.
2	Last Descriptor (LD)	When set to 1, this bit indicates that the buffers pointed to by this descriptor are the last buffers of the data.
1	Disable Interrupt on Completion (DIC)	When set to 1, this bit prevents the setting of the TI/RI bit of the internal DMA controller status register (`idsts`) for the data that ends in the buffer pointed to by this descriptor.
0	Reserved	—

Table 14-7: Internal DMA Controller DES1 Descriptor Field†

The DES1 descriptor field contains the buffer size.

Bits	Name	Description
31:26	Reserved	—
25:13	Buffer 2 Size (BS2)	This field indicates the second data buffer byte size. The buffer size must be a multiple of four. When the buffer size is not a multiple of four, the resulting behavior is undefined. This field is not valid if DES0[4] is set to 1.
12:0	Buffer 1 Size (BS1)	Indicates the data buffer byte size, which must be a multiple of four bytes. When the buffer size is not a multiple of four, the resulting behavior is undefined. If this field is 0, the DMA ignores the buffer and proceeds to the next descriptor for a chain structure, or to the next buffer for a dual-buffer structure. If there is only one descriptor and only one buffer to be programmed, you need to use only buffer 1 and not buffer 2.

Table 14-8: Internal DMA Controller DES2 Descriptor Field†

The DES2 descriptor field contains the address pointer to the data buffer.

Bits	Name	Description
31:0	Buffer Address Pointer 1 (BAP1)	These bits indicate the physical address of the first data buffer. The internal DMA controller ignores DES2 [1:0], because it only performs 32-bit aligned accesses.

Table 14-9: Internal DMA Controller DES3 Descriptor Field†

The DES3 descriptor field contains the address pointer to the next descriptor if the present descriptor is not the last descriptor in a chained descriptor structure or the second buffer address for a dual-buffer structure.†

Bits	Name	Description
31:0	Buffer Address Pointer 2 (BAP2) or Next Descriptor Address	These bits indicate the physical address of the second buffer when the dual-buffer structure is used. If the Second Address Chained (DES0[4]) bit is set to 1, this address contains the pointer to the physical memory where the next descriptor is present. If this is not the last descriptor, the next descriptor address pointer must be aligned to 32 bits. Bits 1 and 0 are ignored.

Host Bus Burst Access

The internal DMA controller attempts to issue fixed-length burst transfers on the master interface if configured using the fixed burst bit (`fb`) of the `bmod` register. The maximum burst length is indicated and limited by the programmable burst length (`pbl`) field of the `bmod` register. When descriptors are being fetched, the master interface always presents a burst size of four to the interconnect.†

The internal DMA controller initiates a data transfer only when sufficient space to accommodate the configured burst is available in the FIFO buffer or the number of bytes to the end of transfer is less than the configured burst-length. When the DMA master interface is configured for fixed-length bursts, it transfers data using the most efficient combination of INCR4, INCR8 or INCR16 and SINGLE transactions. If the DMA master interface is not configured for fixed length bursts, it transfers data using INCR (undefined length) and SINGLE transactions.†

Host Data Buffer Alignment

The transmit and receive data buffers in system memory must be aligned to a 32-bit boundary.

Buffer Size Calculations

The driver knows the amount of data to transmit or receive. For transmitting to the card, the internal DMA controller transfers the exact number of bytes from the FIFO buffer, indicated by the buffer size field of the DES1 descriptor field.†

If a descriptor is not marked as last (with the LD bit of the DES0 field set to 0) then the corresponding buffer(s) of the descriptor are considered full, and the amount of valid data in a buffer is accurately indicated by its buffer size field. If a descriptor is marked as last, the buffer might or might not be full, as indicated by the buffer size in the DES1 field. The driver is aware of the number of locations that are valid.† The driver is expected to ignore the remaining, invalid bytes.

Internal DMA Controller Interrupts

Interrupts can be generated as a result of various events. The `idsts` register contains all the bits that might cause an interrupt. The internal DMA controller interrupt enable register (`idinten`) contains an enable bit for each of the events that can cause an interrupt to occur.†

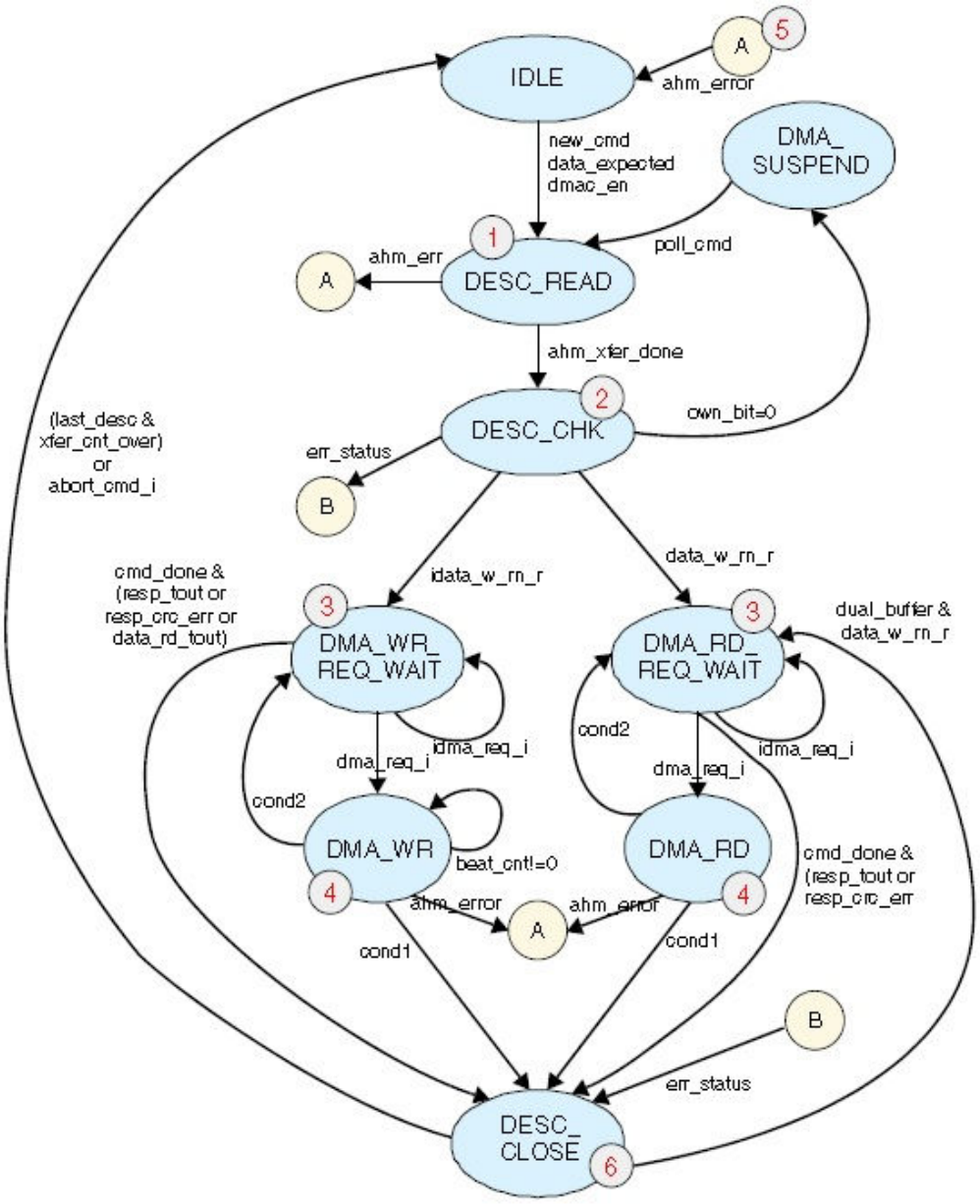
There are two summary interrupts—the normal interrupt summary bit (`nis`) and the abnormal interrupt summary bit (`ais`)—in the `idsts` register.† The `nis` bit results from a logical OR of the transmit interrupt (`ti`) and receive interrupt (`ri`) bits in the `idsts` register. The `ais` bit is a logical OR result of the fatal bus error interrupt (`fbe`), descriptor unavailable interrupt (`du`), and card error summary interrupt (`ces`) bits in the `idsts` register.

Interrupts are cleared by writing a 1 to the corresponding bit position.† If a 0 is written to an interrupt's bit position, the write is ignored, and does not clear the interrupt. When all the enabled interrupts within a group are cleared, the corresponding summary bit is set to 0. When both the summary bits are set to 0, the interrupt signal is de-asserted.†

Interrupts are not queued. If another interrupt event occurs before the driver has responded to the previous interrupt, no additional interrupts are generated. For example, the `ri` bit of the `idsts` register indicates that one or more data has been transferred to the host buffer.†

An interrupt is generated only once for simultaneous, multiple events. The driver must scan the `idsts` register for the interrupt cause.† The final interrupt signal from the controller is a logical OR of the interrupts from the BIU and internal DMA controller.

Internal DMA Controller Functional State Machine†



The following list explains each state of the functional state machine:†

1. The internal DMA controller performs four accesses to fetch a descriptor.†
2. The DMA controller stores the descriptor information internally. If it is the first descriptor, the controller issues a FIFO buffer reset and waits until the reset is complete.†
3. The internal DMA controller checks each bit of the descriptor for the correctness. If bit mismatches are found, the appropriate error bit is set to 1 and the descriptor is closed by setting the OWN bit in the DES0 field to 1.†

The `rintsts` register indicates one of the following conditions:†

- Response timeout†
- Response CRC error†
- Data receive timeout†
- Response error†

4. The DMA waits for the RX watermark to be reached before writing data to system memory, or the TX watermark to be reached before reading data from system memory. The RX watermark represents the number of bytes to be locally stored in the FIFO buffer before the DMA writes to memory. The TX watermark represents the number of free bytes in the local FIFO buffer before the DMA reads data from memory.†
5. If the value of the programmable burst length (PBL) field is larger than the remaining amount of data in the buffer, single transfers are initiated. If dual buffers are being used, and the second buffer contains no data (buffer size = 0), the buffer is skipped and the descriptor is closed.†
6. The OWN bit in descriptor is set to 0 by the internal DMA controller after the data transfer for one descriptor is completed. If the transfer spans more than one descriptor, the DMA controller fetches the next descriptor. If the transfer ends with the current descriptor, the internal DMA controller goes to idle state after setting the `ri` bit or the `ti` bit of the `idsts` register. Depending on the descriptor structure (dual buffer or chained), the appropriate starting address of descriptor is loaded. If it is the second data buffer of dual buffer descriptor, the descriptor is not fetched again.†

Abort During Internal DMA Transfer

If the host issues an SD/SDIO STOP_TRANSMISSION command (CMD12) to the card while data transfer is in progress, the internal DMA controller closes the present descriptor after completing the data transfer until a Data Transfer Over (DTO) interrupt is asserted. Once a STOP_TRANSMISSION command is issued, the DMA controller performs single burst transfers.†

- For a card write operation, the internal DMA controller keeps writing data to the FIFO buffer after fetching it from the system memory until a DTO interrupt is asserted. This is done to keep the card clock running so that the STOP_TRANSMISSION command is reliably sent to the card.†
- For a card read operation, the internal DMA controller keeps reading data from the FIFO buffer and writes to the system memory until a DTO interrupt is generated. This is required because DTO interrupt is not generated until and unless all the FIFO buffer data is emptied.†

Note: For a card write abort, only the current descriptor during which a STOP_TRANSMISSION command is issued is closed by the internal DMA controller. The remaining unread descriptors are not closed by the internal DMA controller.†

Note: For a card read abort, the internal DMA controller reads the data out of the FIFO buffer and writes them to the corresponding descriptor data buffers. The remaining unread descriptors are not closed.†

Fatal Bus Error Scenarios

A fatal bus error occurs when the master interface issues an error response. This error is a system error, so the software driver must not perform any further setup on the controller. The only recovery mechanism from such scenarios is to perform one of the following tasks:†

- Issue a reset to the controller through the reset manager.†
- Issue a program controller reset by writing to the controller reset bit (`controller_reset`) of the `ctrl` register.†

FIFO Buffer Overflow and Underflow

During normal data transfer conditions, FIFO buffer overflow and underflow does not occur. However, if there is a programming error, a FIFO buffer overflow or underflow can result. For example, consider the following scenarios.†

For transmit:†

- PBL=4†
- TX watermark = 1†

For these programming values, if the FIFO buffer has only one location empty, the DMA attempts to read four words from memory even though there is only one word of storage available. This results in a FIFO Buffer Overflow interrupt.†

For receive:†

- PBL=4†
- RX watermark = 1†

For these programming values, if the FIFO buffer has only one location filled, the DMA attempts to write four words, even though only one word is available. This results in a FIFO Buffer Underflow interrupt.†

The driver must ensure that the number of bytes to be transferred, as indicated in the descriptor, is a multiple of four bytes. For example, if the `bytcnt` register = 13, the number of bytes indicated in the descriptor must be rounded up to 16 because the length field must always be a multiple of four bytes.†

PBL and Watermark Levels

This table shows legal PBL and FIFO buffer watermark values for internal DMA controller data transfer operations.†

Table 14-10: PBL and Watermark Levels†

PBL (Number of transfers)	TX/RX FIFO Buffer Watermark Value
1	greater than or equal to 1
4	greater than or equal to 4
8	greater than or equal to 8
16	greater than or equal to 16
32	greater than or equal to 32
64	greater than or equal to 64
128	greater than or equal to 128
256	greater than or equal to 256

CIU

The Card Interface Unit (CIU) interfaces with the BIU and SD/MMC cards or devices. The host processor writes command parameters to the SD/MMC controller's BIU control registers and these parameters are then passed to the CIU. Depending on control register values, the CIU generates SD/MMC command and data traffic on the card bus according to the SD/MMC protocol. The control register values also decide whether the command and data traffic is directed to the CE-ATA card, and the SD/MMC controller controls the command and data path accordingly.†

The following list describes the CIU operation restrictions:†

- After a command is issued, the CIU accepts another command only to check read status or to stop the transfer.†
- Only one data transfer command can be issued at a time.†
- During an open-ended card write operation, if the card clock is stopped because the FIFO buffer is empty, the software must first fill the data into the FIFO buffer and start the card clock. It can then issue only an SD/SDIO STOP_TRANSMISSION (CMD12) command to the card.†
- During an SDIO/COMBO card transfer, if the card function is suspended and the software wants to resume the suspended transfer, it must first reset the FIFO buffer and start the resume command as if it were a new data transfer command.†
- When issuing SD/SDIO card reset commands (GO_IDLE_STATE, GO_INACTIVE_STATE or CMD52_reset) while a card data transfer is in progress, the software must set the stop abort command bit (`stop_abort_cmd`) in the `cmd` register to 1 so that the controller can stop the data transfer after issuing the card reset command.†
- If the card clock is stopped because the FIFO buffer is full during a card read, the software must read at least two FIFO buffer locations to start the card clock.†
- If CE-ATA card device interrupts are enabled (the `nIEN` bit is set to 0 in the ATA control register), a new RW_BLK command must not be sent to the same card device if there is a pending RW_BLK command in progress (the RW_BLK command used in this document is the RW_MULTIPLE_BLOCK MMC command defined by the CE-ATA specification). Only the Command Completion Signal Disable (CCSD) command can be sent while waiting for the Command Completion Signal (CCS).†
- For the same card device, a new command is allowed for reading status information, if interrupts are disabled in the CE-ATA card (the `nIEN` bit is set to 1 in the ATA control register).†
- Open-ended transfers are not supported for the CE-ATA card devices.†
- The `send_auto_stop` signal is not supported (software must not set the `send_auto_stop` bit in the `cmd` register) for CE-ATA transfers.†

The CIU consists of the following primary functional blocks:†

- Command path†
- Data path†
- Clock control†

Command Path

The command path performs the following functions:†

- Load card command parameters†
- Send commands to card bus†
- Receive responses from card bus†
- Send responses to BIU†
- Load clock parameters†
- Drives the P-bit on command pin†

A new command is issued to the controller by writing to the BIU registers and setting the `start_cmd` bit in the `cmd` register. The command path loads the new command (command, command argument, timeout) and sends an acknowledgement to the BIU.†

After the new command is loaded, the command path state machine sends a command to the card bus—including the internally generated seven-term CRC (CRC-7)—and receives a response, if any. The state machine then sends the received response and signals to the BIU that the command is done, and then waits for eight clock cycles before loading a new command. In CE-ATA data payload transfer (RW_MULTIPLE_BLOCK) commands, if the card device interrupts are enabled (the nIEN bit is set to 0

in the ATA control register), the state machine performs the following actions after receiving the response:†

- Does not drive the P-bit; it waits for CCS, decodes and goes back to idle state, and then drives the P-bit.†
- If the host wants to send the CCSD command and if eight clock cycles are expired after the response, it sends the CCSD pattern on the command pin.†

Load Command Parameters

Commands or responses are loaded in the command path in the following situations:†

- New command from BIU—When the BIU sends a new command to the CIU, the `start_cmd` bit is set to 1 in the `cmd` register.†
- Internally-generated `send_auto_stop`—When the data path ends, the SD/SDIO STOP command request is loaded.†
- Interrupt request (IRQ) response with relative card address (RCA) 0x000—When the command path is waiting for an IRQ response from the MMC and a "send irq response" request is signaled by the BIU, the send IRQ request bit (`send_irq_response`) is set to 1 in the `ctrl` register.†

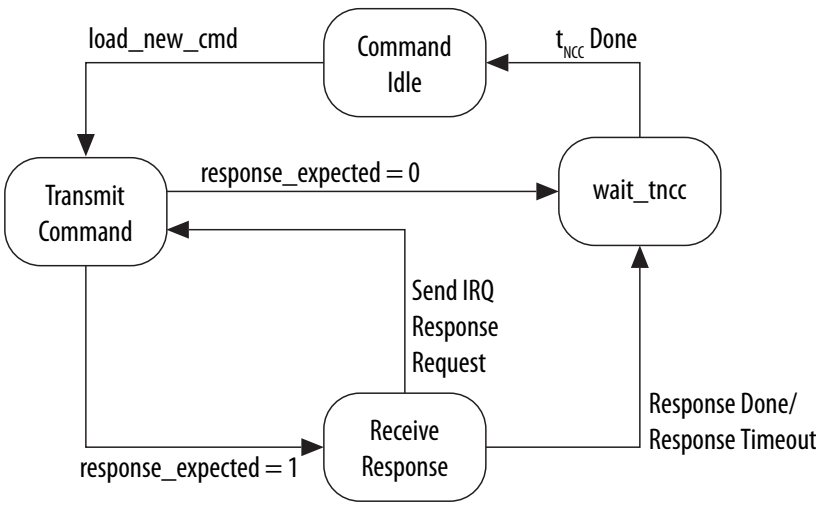
Loading a new command from the BIU in the command path depends on the following `cmd` register bit settings:†

- `update_clock_registers_only`—If this bit is set to 1 in the `cmd` register, the command path updates only the `clkena`, `clkdiv`, and `clksrc` registers. If this bit is set to 0, the command path loads the `cmd`, `cmdarg`, and `tmout` registers. It then processes the new command, which is sent to the card.†
- `wait_prvdata_complete`—If this bit is set to 1, the command path loads the new command under one of the following conditions:†
  - Immediately, if the data path is free (that is, there is no data transfer in progress), or if an open-ended data transfer is in progress (`bytcnt` = 0).†
  - After completion of the current data transfer, if a predefined data transfer is in progress.†

Send Command and Receive Response

After a new command is loaded in the command path (the `update_clock_registers_only` bit in the `cmd` register is set to 0), the command path state machine sends out a command on the card bus.†

Figure 14-6: Command Path State Machine†



The command path state machine performs the following functions, according to cmd register bit values:†

1. send_initialization—Initialization sequence of 80 clock cycles is sent before sending the command.†
2. response_expected—A response is expected for the command. After the command is sent out, the command path state machine receives a 48-bit or 136-bit response and sends it to the BIU. If the start bit of the card response is not received within the number of clock cycles (as set up in the tmout register), the rto bit and command done (CD) bit are set to 1 in the rintsts register, to signal to the BIU. If the response-expected bit is set to 0, the command path sends out a command and signals a response done to the BIU, which causes the cmd bit to be set to 1 in the rintsts register.†
3. response_length—If this bit is set to 1, a 136-bit long response is received; if it is set to 0, a 48-bit short response is received.†
4. check_response_crc—If this bit is set to 1, the command path compares CRC-7 received in the response with the internally-generated CRC-7. If the two do not match, the response CRC error is signaled to the BIU, that is, the rcrc bit is set to 1 in the rintsts register.†

Send Response to BIU

If the response_expected bit is set to 1 in the cmd register, the received response is sent to the BIU. Response register 0 (resp0) is updated for a short response, and the response register 3 (resp3), response register 2 (resp2), response register 1 (resp1), and resp0 registers are updated on a long response, after which the cmd bit is set to 1 in the rintsts register. If the response is for an AUTO_STOP command sent by the CIU, the response is written to the resp1 register, after which the auto command done bit (acd) is set to 1 in the rintsts register.†

The command path verifies the contents of the card response.

Table 14-11: Card Response Fields†

Field	Contents
Response transmission bit	0
Command index	Command index of the sent command
End bit	1

The command index is not checked for a 136-bit response or if the check_response_crc bit in the cmd register is set to 0. For a 136-bit response and reserved CRC 48-bit responses, the command index is reserved, that is, 0b111111.†

Related Information

SD Association

For more information about response values, refer to Physical Layer Simplified Specification, Version 3.01 as described on the SD Association website.

Driving P-bit to the CMD Pin

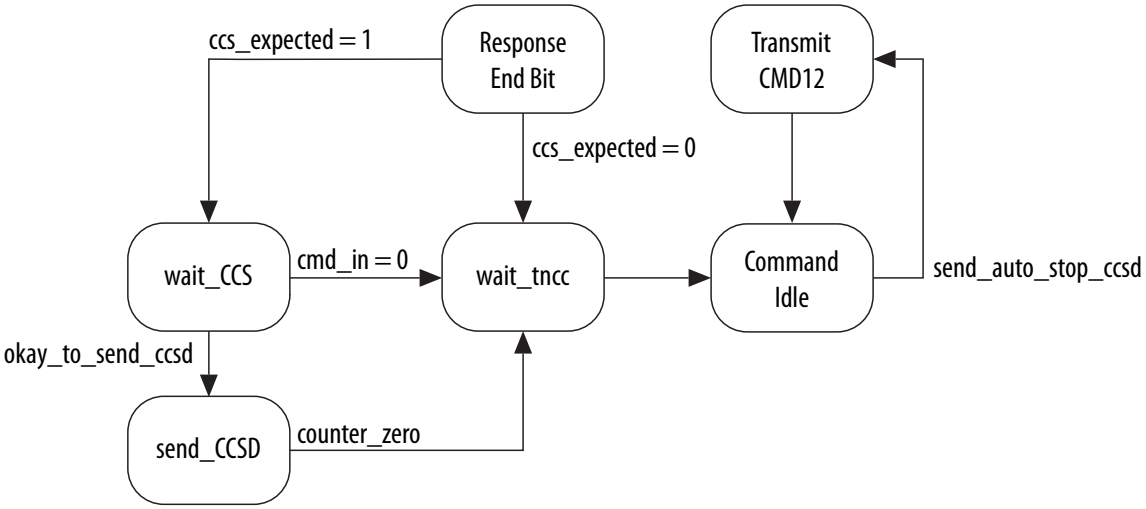
The command path drives a one-cycle pull-up bit (P-bit) to 1 on the CMD pin between two commands if a response is not expected. If a response is expected, the P-bit is driven after the response is received and before the start of the next command. While accessing a CE-ATA card device, for commands that expect a CCS, the P-bit is driven after the response only if the interrupts are disabled in the CE-ATA card (the nIEN

bit is set to 1 in the ATA control register), that is, the CCS expected bit (ccs_expected) in the cmd register is set to 0. If the command expects the CCS, the P-bit is driven only after receiving the CCS.†

Polling the CCS

CE-ATA card devices generate the CCS to notify the host controller of the normal ATA command completion or ATA command termination. After receiving the response from the card, the command path state machine performs the functions illustrated in the following figure according to cmd register bit values.†

Figure 14-7: CE-ATA Command Path State Machine†



The above figure illustrates:

- Response end bit state—The state machine receives the end bit of the response from the card device. If the ccs_expected bit of the cmd register is set to 1, the state machine enters the wait CCS state.†
- Wait CCS—The state machine waits for the CCS from the CE-ATA card device. While waiting for the CCS, the following events can happen:†
  1. Software sets the send CCSD bit (send_ccsd) in the ctrl register, indicating not to wait for CCS and to send the CCSD pattern on the command line.†
  2. Receive the CCS on the CMD line.†
- Send CCSD command—Sends the CCSD pattern (0b00001) on the CMD line.†

CCS Detection and Interrupt to Host Processor

If the ccs_expected bit in the cmd register is set to 1, the CCS from the CE-ATA card device is indicated by setting the data transfer over bit (dto) in the rintsts register. The controller generates a DTO interrupt if this interrupt is not masked.†

For the RW_MULTIPLE_BLOCK commands, if the CE-ATA card device interrupts are disabled (the nIEN bit is set to 1 in the ATA control register)— that is, the ccs_expected bit is set to 0 in the cmd register—there are no CCSs from the card. When the data transfer is over—that is, when the requested number of bytes are transferred—the dto bit in the rintsts register is set to 1.†

CCS Timeout

If the command expects a CCS from the card device (the `ccs_expected` bit is set to 1 in the `cmd` register), the command state machine waits for the CCS and remains in the wait CCS state. If the CE-ATA card fails to send out the CCS, the host software must implement a timeout mechanism to free the command and data path. The controller does not implement a hardware timer; it is the responsibility of the host software to maintain a software timer.†

In the event of a CCS timeout, the host must issue a CCSD command by setting the `send_ccsd` bit in the `ctrl` register. The controller command path state machine sends the CCSD command to the CE-ATA card device and exits to an idle state. After sending the CCSD command, the host must also send an SD/SDIO STOP_TRANSMISSION command to the CE-ATA card to abort the outstanding ATA command.†

Send CCSD Command

If the `send_ccsd` bit in the `ctrl` register is set to 1, the controller sends a CCSD pattern on the CMD line. The host can send the CCSD command while waiting for the CCS or after a CCS timeout happens.†

After sending the CCSD pattern, the controller sets the `cmd` bit in the `rintsts` register and also generates an interrupt to the host if the Command Done interrupt is not masked.†

Note: Within the CIU block, if the `send_ccsd` bit in the `ctrl` register is set to 1 on the same clock cycle as CCS is sampled, the CIU block does not send a CCSD pattern on the CMD line. In this case, the `dto` and `cmd` bits in the `rintsts` register are set to 1.†

Note: Due to asynchronous boundaries, the CCS might have already happened and the `send_ccsd` bit is set to 1. In this case, the CCSD command does not go to the CE-ATA card device and the `send_ccsd` bit is not set to 0. The host must reset the `send_ccsd` bit to 0 before the next command is issued.†

If the send auto stop CCSD (`send_auto_stop_ccsd`) bit in the `ctrl` register is set to 1, the controller sends an internally generated STOP_TRANSMISSION command (CMD12) after sending the CCSD pattern. The controller sets the `acd` bit in the `rintsts` register.†

I/O transmission delay (N_{ACIO} Timeout)

The host software maintains the timeout mechanism for handling the I/O transmission delay (N_{ACIO} cycles) time-outs while reading from the CE-ATA card device. The controller neither maintains any timeout mechanism nor indicates that N_{ACIO} cycles are elapsed while waiting for the start bit of a data token. The I/O transmission delay is applicable for read transfers using the RW_REG and RW_BLK commands; the RW_REG and RW_BLK commands used in this document refer to the RW_MULTIPLE_REGISTER and RW_MULTIPLE_BLOCK MMC commands defined by the CE-ATA specification.†

Note: After the N_{ACIO} timeout, the application must abort the command by sending the CCSD and STOP commands, or the STOP command. The Data Read Timeout (DRTO) interrupt might be set to 1 while a STOP_TRANSMISSION command is transmitted out of the controller, in which case the data read timeout boot data start bit (`bds`) and the `dto` bit in the `rintsts` register are set to 1.†

Data Path

The data path block reads the data FIFO buffer and transmits data on the card bus during a write data transfer, or receives data and writes it to the FIFO buffer during a read data transfer. The data path loads new data parameters—data expected, read/write data transfer, stream/block transfer, block size, byte count, card type, timeout registers—whenever a data transfer command is not in progress. If the data

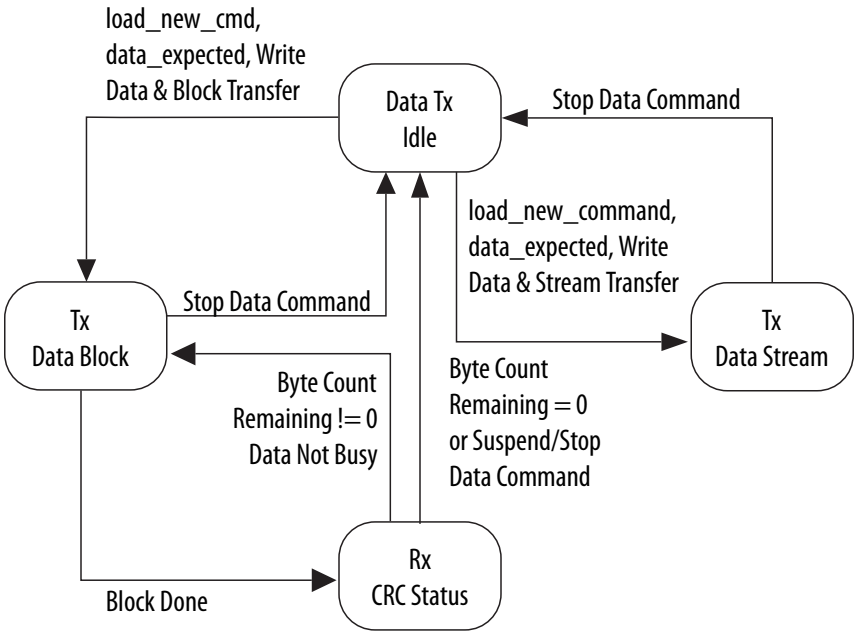
transfer expected bit (`data_expected`) in the `cmd` register is set to 1, the new command is a data transfer command and the data path starts one of the following actions:†

- Transmits data if the read/write bit = 1†
- Receives data if read/write bit = 0†

Data Transmit

The data transmit state machine starts data transmission two clock cycles after a response for the data write command is received. This occurs even if the command path detects a response error or response CRC error. If a response is not received from the card because of a response timeout, data is not transmitted. Depending upon the value of the transfer mode bit (`transfer_mode`) in the `cmd` register, the data transmit state machine puts data on the card data bus in a stream or in blocks.†

Figure 14-8: Data Transmit State Machine†



Stream Data Transmit

If the `transfer_mode` bit in the `cmd` register is set to 1, the transfer is a stream-write data transfer. The data path reads data from the FIFO buffer from the BIU and transmits in a stream to the card data bus. If the FIFO buffer becomes empty, the card clock is stopped and restarted once data is available in the FIFO buffer.†

If the `bytcnt` register is reset to 0, the transfer is an open-ended stream-write data transfer. During this data transfer, the data path continuously transmits data in a stream until the host software issues an SD/SDIO STOP command. A stream data transfer is terminated when the end bit of the STOP command and end bit of the data match over two clock cycles.†

If the `bytcnt` register is written with a nonzero value and the `send_auto_stop` bit in the `cmd` register is set to 1, the STOP command is internally generated and loaded in the command path when the end bit of the STOP command occurs after the last byte of the stream write transfer matches. This data transfer can also terminate if the host issues a STOP command before all the data bytes are transferred to the card bus.†

Single Block Data

If the `transfer_mode` bit in the `cmd` register is set to 0 and the `bytcnt` register value is equal to the value of the `block_size` register, a single-block write-data transfer occurs. The data transmit state machine sends data in a single block, where the number of bytes equals the block size, including the internally-generated 16-term CRC (CRC-16).†

If the `ctype` register is set for a 1-bit, 4-bit, or 8-bit data transfer, the data is transmitted on 1, 4, or 8 data lines, respectively, and CRC-16 is separately generated and transmitted for 1, 4, or 8 data lines, respectively.†

After a single data block is transmitted, the data transmit state machine receives the CRC status from the card and signals a data transfer to the BIU. This happens when the `dto` bit in the `rintsts` register is set to 1.†

If a negative CRC status is received from the card, the data path signals a data CRC error to the BIU by setting the `dcrc` bit in the `rintsts` register.†

Additionally, if the start bit of the CRC status is not received by two clock cycles after the end of the data block, a CRC status start-bit error (SBE) is signaled to the BIU by setting the `sbe` bit in the `rintsts` register.†

Multiple Block Data

A multiple-block write-data transfer occurs if the `transfer_mode` bit in the `cmd` register is set to 0 and the value in the `bytcnt` register is not equal to the value of the `block_size` register. The data transmit state machine sends data in blocks, where the number of bytes in a block equals the block size, including the internally-generated CRC-16 value.†

If the `ctype` register is set to 1-bit, 4-bit, or 8-bit data transfer, the data is transmitted on 1-, 4-, or 8-data lines, respectively, and CRC-16 is separately generated and transmitted on 1-, 4-, or 8-data lines, respectively.†

After one data block is transmitted, the data transmit state machine receives the CRC status from the card. If the remaining byte count becomes 0, the data path signals to the BIU that the data transfer is done. This happens when the `dto` bit in the `rintsts` register is set to 1.†

If the remaining data bytes are greater than zero, the data path state machine starts to transmit another data block.†

If a negative CRC status is received from the card, the data path signals a data CRC error to the BIU by setting the `dcrc` bit in the `rintsts` register, and continues further data transmission until all the bytes are transmitted.†

If the CRC status start bit is not received by two clock cycles after the end of a data block, a CRC status SBE is signaled to the BIU by setting the `ebe` bit in the `rintsts` register and further data transfer is terminated.†

If the `send_auto_stop` bit is set to 1 in the `cmd` register, the SD/SDIO STOP command is internally generated during the transfer of the last data block, where no extra bytes are transferred to the card. The end bit of the STOP command might not exactly match the end bit of the CRC status in the last data block.†

If the block size is less than 4, 16, or 32 for card data widths of 1 bit, 4 bits, or 8 bits, respectively, the data transmit state machine terminates the data transfer when all the data is transferred, at which time the internally-generated STOP command is loaded in the command path.†

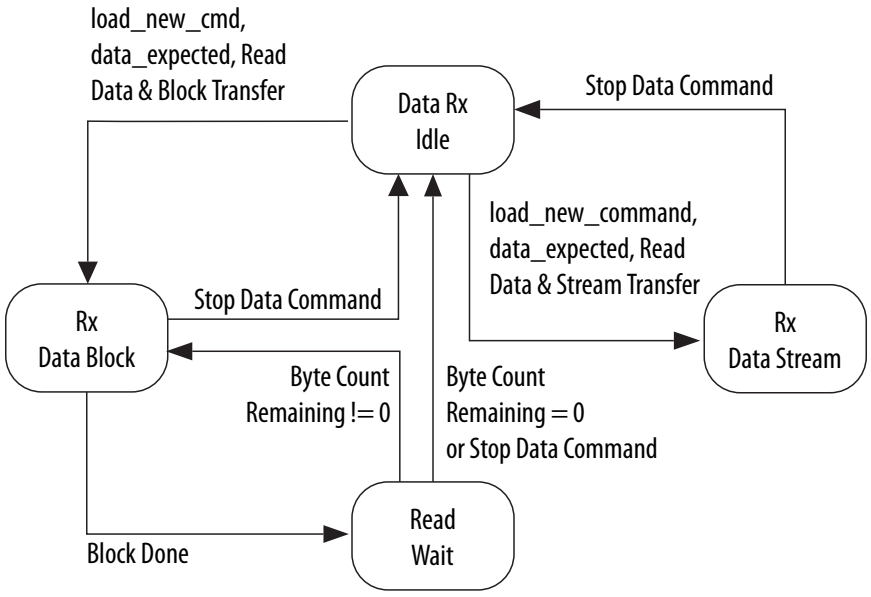
If the bytcnt is zero (the block size must be greater than zero) the transfer is an open-ended block transfer. The data transmit state machine for this type of data transfer continues the block-write data transfer until the host software issues an SD/SDIO STOP or STOP_TRANSMISSION (CMD12) command.†

Data Receive

The data-receive state machine receives data two clock cycles after the end bit of a data read command, even if the command path detects a response error or response CRC error. If a response is not received from the card because a response timeout occurs, the BIU does not receive a signal that the data transfer is complete. This happens if the command sent by the controller is an illegal operation for the card, which keeps the card from starting a read data transfer.†

If data is not received before the data timeout, the data path signals a data timeout to the BIU and an end to the data transfer done. Based on the value of the transfer_mode bit in the cmd register, the data-receive state machine gets data from the card data bus in a stream or block(s).†

Figure 14-9: Data Receive State Machine†



Stream Data Read

A stream-read data transfer occurs if the transfer_mode bit in the cmd register is set to 1, at which time the data path receives data from the card and writes it to the FIFO buffer. If the FIFO buffer becomes full, the card clock stops and restarts once the FIFO buffer is no longer full.†

An open-ended stream-read data transfer occurs if the bytcnt register is set to 0. During this type of data transfer, the data path continuously receives data in a stream until the host software issues an SD/SDIO STOP command. A stream data transfer terminates two clock cycles after the end bit of the STOP command.†

If the bytcnt register contains a nonzero value and the send_auto_stop bit in the cmd register is set to 1, a STOP command is internally generated and loaded into the command path, where the end bit of the STOP command occurs after the last byte of the stream data transfer is received. This data transfer can

terminate if the host issues an SD/SDIO STOP or STOP_TRANSMISSION (CMD12) command before all the data bytes are received from the card.†

Single-block Data Read

If the `ctype` register is set to a 1-bit, 4-bit, or 8-bit data transfer, data is received from 1, 4, or 8 data lines, respectively, and CRC-16 is separately generated and checked for 1, 4, or 8 data lines, respectively. If there is a CRC-16 mismatch, the data path signals a data CRC error to the BIU. If the received end bit is not 1, the BIU receives an End-bit Error (EBE).†

Multiple-block Data Read

If the `transfer_mode` bit in the `cmd` register is clear and the value of the `bytcnt` register is not equal to the value of the `block_size` register, the transfer is a multiple-block read-data transfer. The data-receive state machine receives data in blocks, where the number of bytes in a block is equal to the block size, including the internally-generated CRC-16.†

If the `ctype` register is set to a 1-bit, 4-bit, or 8-bit data transfer, data is received from 1, 4, or 8 data lines, respectively, and CRC-16 is separately generated and checked for 1, 4, or 8 data lines, respectively. After a data block is received, if the remaining byte count becomes zero, the data path signals a data transfer to the BIU.†

If the remaining data bytes are greater than zero, the data path state machine causes another data block to be received. If CRC-16 of a received data block does not match the internally-generated CRC-16, a data CRC error to the BIU and data reception continue further data transmission until all bytes are transmitted. Additionally, if the end of a received data block is not 1, data on the data path signals terminate the bit error to the CIU and the data-receive state machine terminates data reception, waits for data timeout, and signals to the BIU that the data transfer is complete.†

If the `send_auto_stop` bit in the `cmd` register is set to 1, the SD/SDIO STOP command is internally generated when the last data block is transferred, where no extra bytes are transferred from the card. The end bit of the STOP command might not exactly match the end bit of the last data block.†

If the requested block size for data transfers to cards is less than 4, 16, or 32 bytes for 1-bit, 4-bit, or 8-bit data transfer modes, respectively, the data-transmit state machine terminates the data transfer when all data is transferred, at which point the internally-generated STOP command is loaded in the command path. Data received from the card after that are then ignored by the data path.†

If the `bytcnt` register is 0 (the block size must be greater than zero), the transfer is an open-ended block transfer. For this type of data transfer, the data-receive state machine continues the block-read data transfer until the host software issues an SD/SDIO STOP or STOP_TRANSMISSION (CMD12) command.†

Auto-Stop

The controller internally generates an SD/SDIO STOP command and is loaded in the command path when the `send_auto_stop` bit in the `cmd` register is set to 1. The AUTO_STOP command helps to send an exact number of data bytes using a stream read or write for the MMC, and a multiple-block read or write for SD memory transfer for SD cards. The software must set the `send_auto_stop` bit according to the following details: †

The following list describes conditions for the AUTO_STOP command:†

- Stream-read for MMC with byte count greater than zero—The controller generates an internal STOP command and loads it into the command path so that the end bit of the STOP command is sent when the last byte of data is read from the card and no extra data byte is received. If the byte count is less than six (48 bits), a few extra data bytes are received from the card before the end bit of the STOP command is sent.†
- Stream-write for MMC with byte count greater than zero—The controller generates an internal STOP command and loads it into the command path so that the end bit of the STOP command is sent when the last byte of data is transmitted on the card bus and no extra data byte is transmitted. If the byte count is less than six (48 bits), the data path transmits the data last to meet these condition.†
- Multiple-block read memory for SD card with byte count greater than zero—If the block size is less than four (single-bit data bus), 16 (4-bit data bus), or 32 (8-bit data bus), the AUTO_STOP command is loaded in the command path after all the bytes are read. Otherwise, the STOP command is loaded in the command path so that the end bit of the STOP command is sent after the last data block is received.†
- Multiple-block write memory for SD card with byte count greater than zero—If the block size is less than three (single-bit data bus), 12 (4-bit data bus), or 24 (8-bit data bus), the AUTO_STOP command is loaded in the command path after all data blocks are transmitted. Otherwise, the STOP command is loaded in the command path so that the end bit of the STOP command is sent after the end bit of the CRC status is received.†
- Precaution for host software during auto-stop—When an AUTO_STOP command is issued, the host software must not issue a new command to the controller until the AUTO_STOP command is sent by the controller and the data transfer is complete. If the host issues a new command during a data transfer with the AUTO_STOP command in progress, an AUTO_STOP command might be sent after the new command is sent and its response is received. This can delay sending the STOP command, which transfers extra data bytes. For a stream write, extra data bytes are erroneous data that can corrupt the card data. If the host wants to terminate the data transfer before the data transfer is complete, it can issue an SD/SDIO STOP or STOP_TRANSMISSION (CMD12) command, in which case the controller does not generate an AUTO_STOP command.†

Auto-Stop Generation for MMC Cards

Table 14-12: Auto-Stop Generation for MMC Cards†

Transfer Type	Byte Count	send_auto_stop bit set	Comments
Stream read	0	No	Open-ended stream
Stream read	>0	Yes	Auto-stop after all bytes transfer
Stream write	0	No	Open-ended stream
Stream write	>0	Yes	Auto-stop after all bytes transfer
Single-block read	>0	No	Byte count = 0 is illegal
Single-block write	>0	No	Byte count = 0 is illegal
Multiple-block read	0	No	Open-ended multiple block
Multiple-block read	>0	Yes (40) †	Pre-defined multiple block

(40) The condition under which the transfer mode is set to block transfer and byte_count is equal to block size is treated as a single-block data transfer command for both MMC and SD cards. If byte_count = n*block_size (n = 2, 3, …), the condition is treated as a predefined multiple-block data transfer command. In the case of

Transfer Type	Byte Count	send_auto_stop bit set	Comments
Multiple-block write	0	No	Open-ended multiple block
Multiple-block write	>0	Yes (40)†	Pre-defined multiple block

Auto-Stop Generation for SD Cards

Table 14-13: Auto-Stop Generation for SD Cards†

Transfer Type	Byte Count	send_auto_stop bit set	Comments
Single-block read	>0	No	Byte count = 0 is illegal
Single-block write	>0	No	Byte count = 0 illegal
Multiple-block read	0	No	Open-ended multiple block
Multiple-block read	>0	Yes	Auto-stop after all bytes transfer
Multiple-block write	0	No	Open-ended multiple block
Multiple-block write	>0	Yes	Auto-stop after all bytes transfer

Auto-Stop Generation for SDIO Cards

Table 14-14: Auto-Stop Generation for SDIO Cards†

Transfer Type	Byte Count	send_auto_stop bit set	Comments
Single-block read	>0	No	Byte count = 0 is illegal
Single-block write	>0	No	Byte count = 0 illegal
Multiple-block read	0	No	Open-ended multiple block
Multiple-block read	>0	No	Pre-defined multiple block
Multiple-block write	0	No	Open-ended multiple block
Multiple-block write	>0	No	Pre-defined multiple block

Non-Data Transfer Commands that Use Data Path

Some SD/SDIO non-data transfer commands (commands other than read and write commands) also use the data path.

---

an MMC card, the host software can perform a predefined data transfer in two ways: 1) Issue the CMD23 command before issuing CMD18/CMD25 commands to the card – in this case, issue CMD18/CMD25 commands without setting the send_auto_stop bit. 2) Issue CMD18/CMD25 commands without issuing CMD23 command to the card, with the send_auto_stop bit set. In this case, the multiple-block data transfer is terminated by an internally-generated auto-stop command after the programmed byte count.†

Table 14-15: Non-Data Transfer Commands and Requirements†

	PROGRAM_CSD (CMD27)	SEND_WRITE_PROT (CMD30)	LOCK_UNLOCK (CMD42)	SD_STATUS (ACMD13)	SEND_NUM_WR_BLOCKS (ACMD22)	SEND_SCR (ACMD51)
Command register programming†						
Cmd_index	0x1B=27	0x1E=30	0x2A=42	0x0D=13	0x16=22	0x33=51
Response_expect†	1	1	1	1	1	1
Response_length†	0	0	0	0	0	0
Check_response_crc†	1	1	1	1	1	1
Data_expected†	1	1	1	1	1	1
Read/write†	1	0	1	0	0	0
Transfer_mode†	0	0	0	0	0	0
Send_auto_stop†	0	0	0	0	0	0
Wait_prevdata_complete†	0	0	0	0	0	0
Stop_abort_cmd†	0	0	0	0	0	0

Table 14-16: Non-Data Transfer Commands and Requirements (Cont.)†

	PROGRAM_CSD (CMD27)	SEND_WRITE_PROT (CMD30)	LOCK_UNLOCK (CMD42)	SD_STATUS (ACMD13)	SEND_NUM_WR_BLOCKS (ACMD22)	SEND_SCR (ACMD51)
Command Argument register programming†						
	Stuff bits	32-bit write protect data address	Stuff bits	Stuff bits	Stuff bits	Stuff bits

Table 14-17: Non-Data Transfer Commands and Requirements†

PROGRAM_CSD (CMD27)	SEND_WRITE_ PROT (CMD30)	LOCK_UNLOCK (CMD42)	SD_ STATUS (ACMD13)	SEND_NUM_WR_ BLOCKS (ACMD22)	SEND_SCR (ACMD51)
Block Size register programing†					
16	4	Num_bytes(41)	64	4	8

Table 14-18: Non-Data Transfer Commands and Requirements†

PROGRAM_CSD (CMD27)	SEND_WRITE_ PROT (CMD30)	LOCK_UNLOCK (CMD42)	SD_ STATUS (ACMD13)	SEND_NUM_WR_ BLOCKS (ACMD22)	SEND_SCR (ACMD51)
Byte Count register programming†					
16	4	Num_bytes(42)	64	4	8

Related Information

- **SD Association**
  For more information, the SD specification can be purchased from this organization.
- **JEDEC Global Standards of the Microelectronics Industry**
  For more information, the MMC specification can be purchased from this organization.

Clock Control Block

The clock control block provides different clock frequencies required for SD/MMC/CE-ATA cards. The clock control block has one clock divider, which is used to generate different card clock frequencies.†

The clock frequency of a card depends on the following clock `ctrl` register settings:†

- `clkdiv` register—Internal clock dividers are used to generate different clock frequencies required for the cards. The division factor for the clock divider can be set by writing to the `clkdiv` register. The clock divider is an 8-bit value that provides a clock division factor from 1 to 510; a value of 0 represents a clock-divider bypass, a value of 1 represents a divide by 2, a value of 2 represents a divide by 4, and so on.†
- `clksrc` register—Set this register to 0 as clock is divided by clock divider 0.†
- `clkena` register—The `cclk_out` card output clock can be enabled or disabled under the following conditions:†
  - `cclk_out` is enabled when the `cclk_enable` bit in the `clkena` register is set to 1 and disabled when set to 0.†
  - Low-power mode can be enabled by setting the `cclk_low_power` bit of the `clkena` register to 1. If low-power mode is enabled to save card power, the `cclk_out` signal is disabled when the card is idle for at least eight card clock cycles. Low-power mode is enabled when a new command is loaded and the command path goes to a non-idle state.†

Under the following conditions, the card clock is stopped or disabled:†

(41) Num_bytes = Number of bytes specified as per the lock card data structure. Refer to the SD specification and the MMC specification.†

(42) Num_bytes = Number of bytes specified as per the lock card data structure. Refer to the SD specification and the MMC specification.†

- Clock can be disabled by writing to the `clkena` register.†
- When low-power mode is selected and the card is idle for at least eight clock cycles.†
- FIFO buffer is full, data path cannot accept more data from the card, and data transfer is incomplete—to avoid FIFO buffer overflow.†
- FIFO buffer is empty, data path cannot transmit more data to the card, and data transfer is incomplete —to avoid FIFO buffer underflow.†

Note: The card clock must be disabled through the `clkena` register before the host software changes the values of the `clkdiv` and `clksrc` registers.†

Error Detection

Errors can occur during card operations within the CIU in the following situations.

Response†

- Response timeout—did not receive the response expected with response start bit within the specified number of clock cycles in the timeout register.†
- Response CRC error—response is expected and check response CRC requested; response CRC-7 does not match with the internally-generated CRC-7.†
- Response error—response transmission bit is not 0, command index does not match with the command index of the send command, or response end bit is not 1.†

Data Transmit†

- No CRC status—during a write data transfer, if the CRC status start bit is not received for two clock cycles after the end bit of the data block is sent out, the data path performs the following actions:†
  - Signals no CRC status error to the BIU†
  - Terminates further data transfer†
  - Signals data transfer done to the BIU†
- Negative CRC—if the CRC status received after the write data block is negative (that is, not 0b010), the data path signals a data CRC error to the BIU and continues with the data transfer.†
- Data starvation due to empty FIFO buffer—if the FIFO buffer becomes empty during a write data transmission, or if the card clock stopped and the FIFO buffer remains empty for a data-timeout number of clock cycles, the data path signals a data-starvation error to the BIU and the data path continues to wait for data in the FIFO buffer.†

Data Receive

- Data timeout—during a read-data transfer, if the data start bit is not received before the number of clock cycles specified in the timeout register, the data path does the following action: †
  - Signals a data-timeout error to the BIU†
  - Terminates further data transfer†
  - Signals data transfer done to BIU†
- Data SBE—during a 4-bit or 8-bit read-data transfer, if the all-bit data line does not have a start bit, the data path signals a data SBE to the BIU and waits for a data timeout, after which it signals that the data transfer is done.†
- Data CRC error—during a read-data-block transfer, if the CRC-16 received does not match with the internally generated CRC-16, the data path signals a data CRC error to the BIU and continues with the data transfer.†
- Data EBE—during a read-data transfer, if the end bit of the received data is not 1, the data path signals an EBE to the BIU, terminates further data transfer, and signals to the BIU that the data transfer is done.†
- Data starvation due to FIFO buffer full—during a read data transmission and when the FIFO buffer becomes full, the card clock stops. If the FIFO buffer remains full for a data-timeout number of clock cycles, the data path signals a data starvation error to the BIU, by setting the data starvation host timeout bit (`hto`) in `rintsts` register to 1, and the data path continues to wait for the FIFO buffer to empty.†

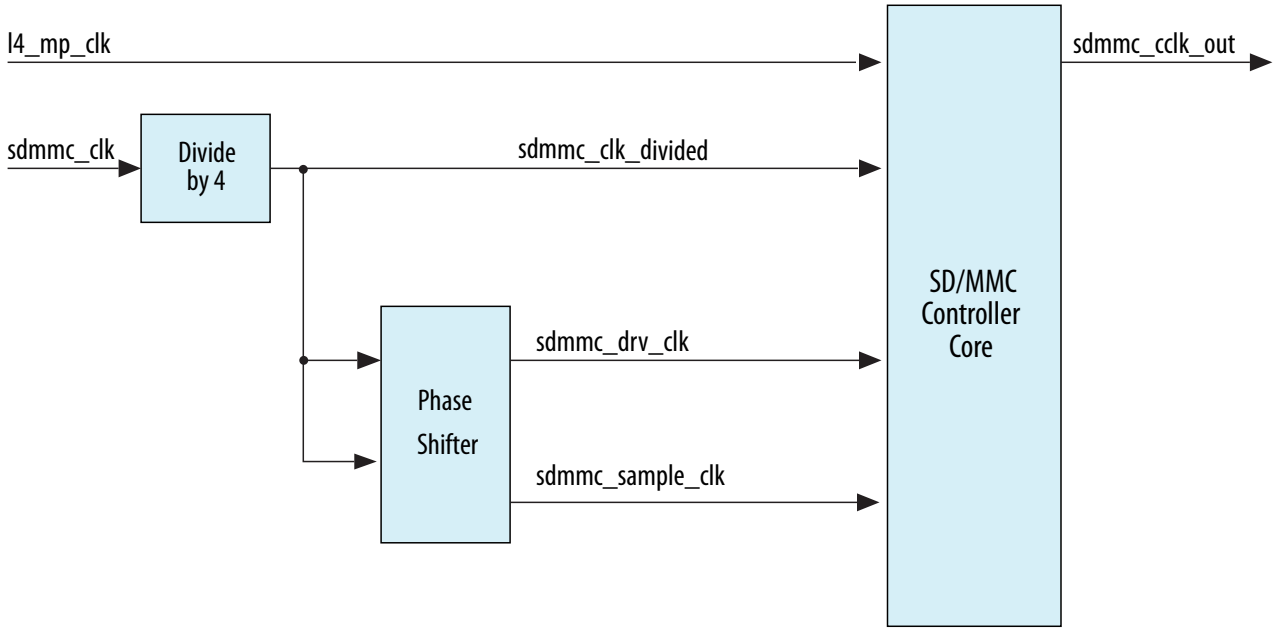
Clocks

Table 14-19: SD/MMC Controller Clocks

Clock Name	Direction	Description
`l4_mp_clk`	In	Clock for SD/MMC controller BIU
`sdmmc_clk`	In	Clock for SD/MMC controller
`sdmmc_cclk_out`	Out	Generated output clock for card
`sdmmc_clk_divided`	Internal	Divide-by-four clock of `sdmmc_clk`
`sdmmc_sample_clk`	Internal	Phase-shifted clock of `sdmmc_clk_divided` used to sample the command and data from the card
`sdmmc_drv_clk`	Internal	Phase-shifted clock of `sdmmc_clk_divided` for controller to drive command and data to the card to meet hold time requirements

Figure 14-10: SD/MMC Controller Clock Connections - HPS

This figure displays the controller clock connections from the HPS.



The `sdmmc_clk` clock from the clock manager is divided by four and becomes the `sdmmc_clk_divided` clock before passing to the phase shifters and the SD/MMC controller CIU. The phase shifters are used to generate the `sdmmc_drv_clk` and `sdmmc_sample_clk` clocks. These phase shifters provide up to eight phases shift which include 0, 45, 90, 135, 180, 225, 270, and 315 degrees. The `sdmmc_sample_clk` clock can be driven by the output from the phase shifter.

Note: The selections of phase shift degree and `sdmmc_sample_clk` source are done in the system manager. For information about setting the phase shift and selecting the source of the `sdmmc_sample_clk` clock, refer to the "Clock Setup" section within this document.

The controller generates the `sdmmc_cclk_out` clock, which is driven to the card. For more information about the generation of the `sdmmc_cclk_out` clock, refer to the "Clock Control Block" section within this document.

Related Information

- **Clock Setup** on page 14-46
  Refer to this section for information about setting the phase shift.
- **Clock Control Block** on page 14-33
  Refer to this section for information about the generation of the `sdmmc_cclk_out`clock.

Resets

The SD/MMC controller has one reset signal. The reset manager drives this signal to the SD/MMC controller on a cold or warm reset.

Related Information

Reset Manager on page 3-1

Taking the SD/MMC Controller Out of Reset

When a cold or warm reset is issued in the HPS, the reset manager resets this module and holds it in reset until software releases it.

After the Cortex-A9 MPCore CPU boots, it can deassert the reset signal by clearing the appropriate bits in the reset manager's corresponding reset register. For details about reset registers, refer to "Module Reset Signals".

Related Information

Module Reset Signals on page 3-8

Voltage Switching

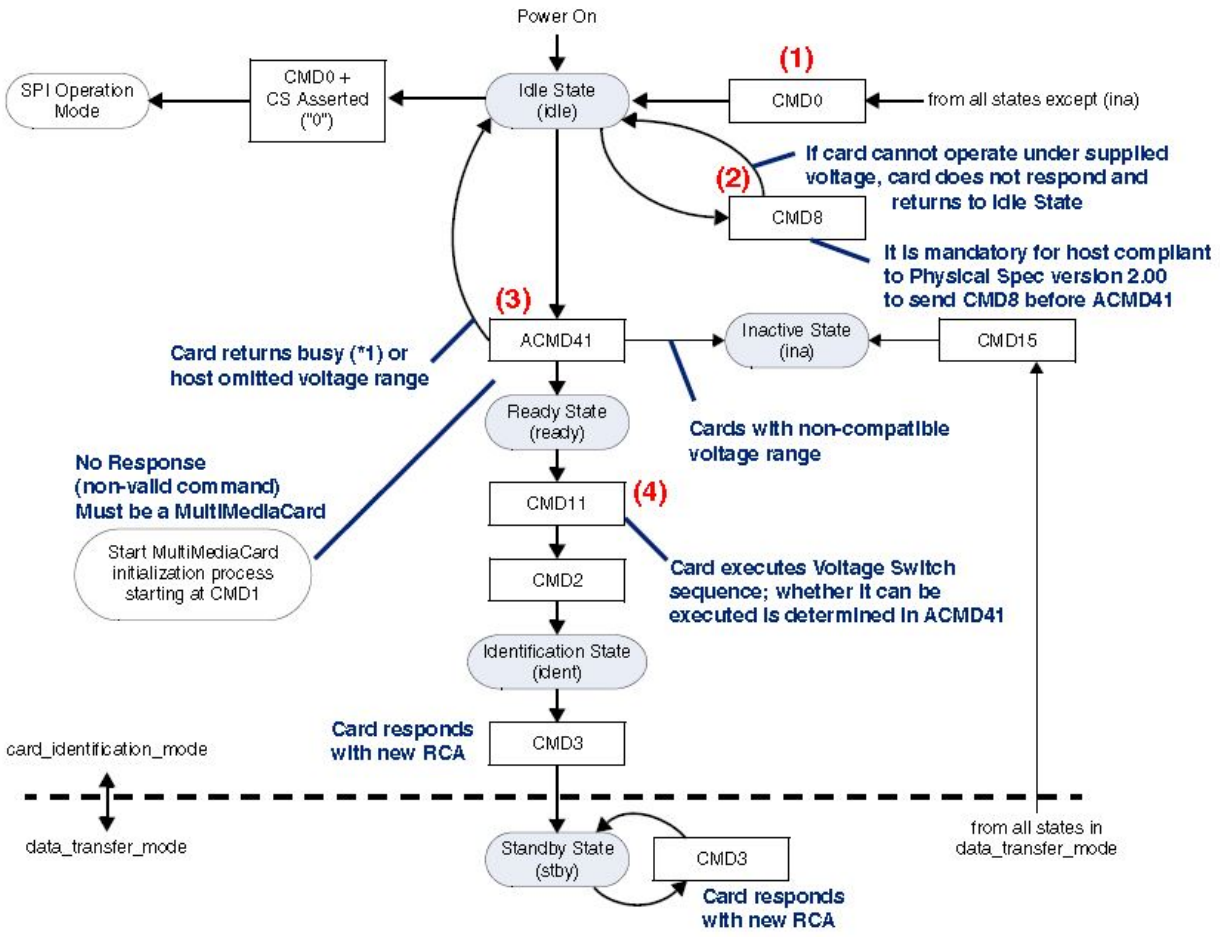
This section describes the general steps to switch voltage level.

The SD/MMC cards support various operating voltages, for example 1.8V and 3.0V. If you have a card which is at 1.8V and you eject it and replace it with another card, which is 3.0V, then voltage switching is required.

Many SD cards have an option to signal at 1.8 or 3.0 V, however the initial power-up voltage requirement is 3.0V. To support these different voltage requirements, external transceivers are needed.

The general steps to switch the voltage level requires you to use a SD/MMC voltage-translation transceiver in between the HPS and the SD/MMC card.

Figure 14-11: Voltage Switching Command Flow Diagram†



(*1) **Note:** Card returns busy when:†

- Card executes internal initializaiton process†
- Card is High or Extended capacity SD Memory Card and host does not support High†

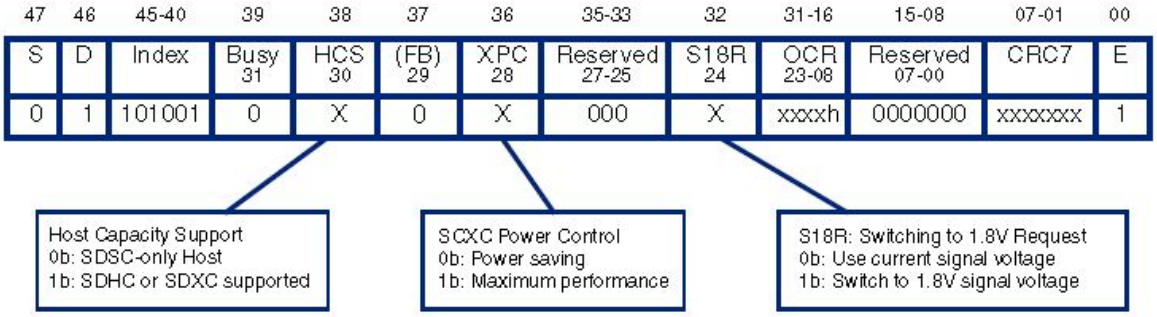
The following outlines the steps for the voltage switch programming sequence.†

1. Software Driver starts CMD0, which selects the bus mode as SD.†
2. After the bus is in SD card mode, CMD8 is started in order to verify if the card is compatible with the SD Memory Card Specification, Version 2.00.†

   CMD8 determines if the card is capable of working within the host supply voltage specified in the VHS (19:16) field of the CMD; the card supports the current host voltage if a response to CMD8 is received.†
3. ACMD 41 is started.†

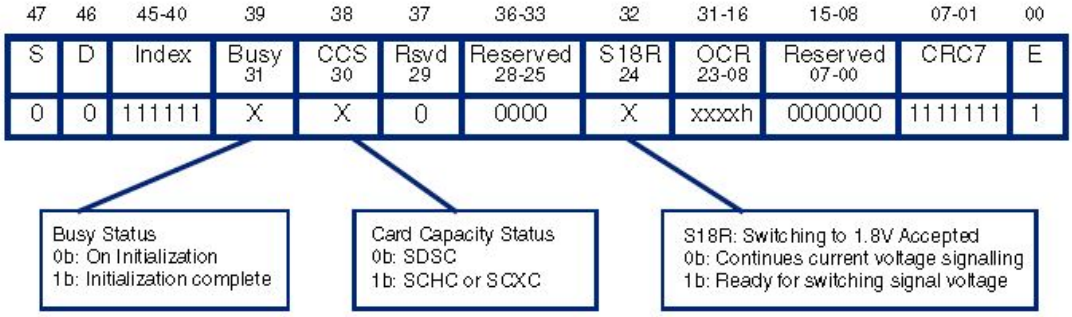
   The response to this command informs the software if the card supports voltage switching; bits 38, 36, and 32 are checked by the card argument of ACMD41.†

Figure 14-12: ACMD41 Argument



a. Bit 30 informs the card if host supports SDHC/SDXC or not; this bit should be set to 1'b1.†
b. Bit 28 can be either 1 or 0.†
c. Bit 24 should be set to 1'b1, indicating that the host is capable of voltage switching.†

Figure 14-13: ACMD41 Response (R3)†



d. Bit 30 – If set to 1'b1, card supports SDHC/SDXC; if set to 1'b0, card supports only SDSC.†
e. Bit 24 – If set to 1'b1, card supports voltage switching and is ready for the switch.†
f. Bit 31 – If set to 1'b1, initialization is over; if set to 1'b0, means initialization in process†

4. If the card supports voltage switching, then the software must perform the steps discussed for either the "Voltage Switch Normal Scenario" or the "Voltage Switch Error Scenario", located in the *Synopsys DesignWare Cores Mobile Storage Host Databook*.

Related Information

Synopsys DesignWare Cores Mobile Storage Host Databook
For more information about Voltage Switching

SD/MMC Controller Programming Model

Software and Hardware Restrictions†

Only one data transfer command should be issued at one time. For CE-ATA devices, if CE-ATA device interrupts are enabled (nIEN=0), only one RW_MULTIPLE_BLOCK command (RW_BLK) should be

issued; no other commands (including a new RW_BLK) should be issued before the Data Transfer Over status is set for the outstanding RW_BLK.†

Before issuing a new data transfer command, the software should ensure that the card is not busy due to any previous data transfer command. Before changing the card clock frequency, the software must ensure that there are no data or command transfers in progress.†

If the card is enumerated in SDR12 or SDR25 mode, the application must program the `use_hold_reg` bit[29] in the CMD register to 1'b1.†

This programming should be done for all data transfer commands and non-data commands that are sent to the card. When the `use_hold_reg` bit is programmed to 1'b0, the SD/MMC controller bypasses the Hold Registers in the transmit path. The value of this bit should not be changed when a Command or Data Transfer is in progress.†

For more information on using the `use_hold_reg` and the implementation requirements for meeting the card input hold time, refer to the latest version of the Synopsys DesignWare Cores Mobile Storage Host Databook.

Avoiding Glitches in the Card Clock Outputs†

To avoid glitches in the card clock outputs (`sdmmc_cclk_out`), the software should use the following steps when changing the card clock frequency:†

1. Before disabling the clocks, ensure that the card is not busy due to any previous data command. To determine this, check for 0 in bit 9 of the STATUS register.†
2. Update the Clock Enable register to disable all clocks. To ensure completion of any previous command before this update, send a command to the CIU to update the clock registers by setting:†
   - `start_cmd` bit†
   - "update clock registers only" bits†
   - "wait_previous data complete"†

   Note: Wait for the CIU to take the command by polling for 0 on the `start_cmd` bit.†
3. Set the `start_cmd` bit to update the Clock Divider and/or Clock Source registers, and send a command to the CIU in order to update the clock registers; wait for the CIU to take the command.†
4. Set `start_cmd` to update the Clock Enable register in order to enable the required clocks and send a command to the CIU to update the clock registers; wait for the CIU to take the command.†

Reading from a Card in Non-DMA Mode†

When a card is read in non-DMA mode, the Data Transfer Over (RINTSTS[3]) interrupt occurs as soon as the data transfer from the card is over. There still could be some data left in the FIFO, and the RX_WMark interrupt may or may not occur, depending on the remaining bytes in the FIFO. Software should read any remaining bytes upon seeing the Data Transfer Over (DTO) interrupt. While using the external DMA interface for reading from a card, the DTO interrupt occurs only after all the data is flushed to memory by the DMA Interface unit.†

Writing to a Card in External DMA Mode†

While writing to a card in external DMA mode, if an undefined-length transfer is selected by setting the Byte Count register to 0, the DMA logic will likely request more data than it will send to the card, since it has no way of knowing at which point the software will stop the transfer. The DMA request stops as soon as the DTO is set by the CIU.†

Software Issues a Controller_Reset Command†

If the software issues a controller_reset command by setting control register bit[0] to 1, all the CIU state machines are reset; the FIFO is not cleared. The DMA sends all remaining bytes to the host. In addition to a card-reset, if a FIFO reset is also issued, then:†

- Any pending DMA transfer on the bus completes correctly†
- DMA data read is ignored†
- Write data is unknown (x)†

Additionally, if dma_reset is also issued, any pending DMA transfer is abruptly terminated. When the DW-DMA/Non-DW-DMA is used, the DMA controller channel should also be reset and reprogrammed.†

If any of the previous data commands do not properly terminate, then the software should issue the FIFO reset in order to remove any residual data, if any, in the FIFO. After asserting the FIFO reset, you should wait until this bit is cleared.†

Data-Transfer Requirement Between the FIFO and Host†

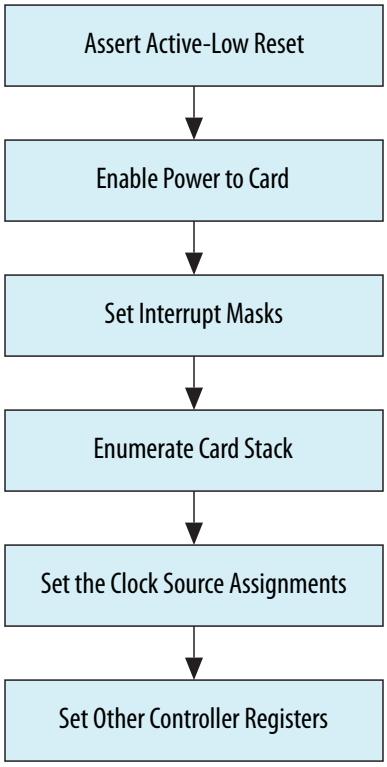
One data-transfer requirement between the FIFO and host is that the number of transfers should be a multiple of the FIFO data width (F_DATA_WIDTH). For example, if F_DATA_WIDTH = 32 and you want to write only 15 bytes to an SD_MMC_CEATA card (BYTCNT), the host should write 16 bytes to the FIFO or program the DMA to do 16-byte transfers, if external DMA mode is enabled. The software can still program the Byte Count register to only 15, at which point only 15 bytes will be transferred to the card. Similarly, when 15 bytes are read from a card, the host should still read all 16 bytes from the FIFO.†

It is recommended that you not change the FIFO threshold register in the middle of data transfers when DW-DMA/Non-DW-DMA mode is chosen.†

Initialization†

After the power and clock to the controller are stable, the controller active-low reset is asserted. The reset sequence initializes the registers, FIFO buffer pointers, DMA interface controls, and state machines in the controller.†

Figure 14-14: SD/MMC Controller Initialization Sequence†



Power-On Reset Sequence

Software must perform the following steps after the power-on-reset:

1. Before enabling power to the card, confirm that the voltage setting to the voltage regulator is correct. †
2. Enable power to the card by setting the power enable bit (`power_enable`) in the power enable register (`pwren`) to 1. Wait for the power ramp-up time before proceeding to the next step.†
3. Set the interrupt masks by resetting the appropriate bits to 0 in the `intmask` register.†
4. Set the `int_enable` bit of the `ctrl` register to 1.†

   Note: Altera recommends that you write 0xFFFFFFFF to the `rintsts` register to clear any pending interrupts before setting the `int_enable` bit to 1.†
5. Discover the card stack according to the card type. For discovery, you must restrict the clock frequency to 400 kHz in accordance with SD/MMC/CE-ATA standards. For more information, refer to *Enumerated Card Stack*.†
6. Set the clock source assignments. Set the card frequency using the `clkdiv` and `clksrc` registers of the controller. For more information, refer to *Clock Setup*.†
7. The following common registers and fields can be set during initialization process:†

- The response timeout field (`response_timeout`) of the `tmout` register. A typical value is `0x40`.†
- The data timeout field (`data_timeout`) of the `tmout` register, highest of the following:†
  - $10 * N_{AC}$ †

    N_{AC} = card device total access time†

    $= 10 * ((TAAC * F_{OP}) + (100 * NSAC))$†

    where:†

    TAAC = Time-dependent factor of the data access time†

    F_{OP} = The card clock frequency used for the card operation†

    NSAC = Worst-case clock rate-dependent factor of the data access time†
  - Host FIFO buffer latency†

    On read: Time elapsed before host starts reading from a full FIFO buffer†

    On write: Time elapsed before host starts writing to an empty FIFO buffer†
- Debounce counter register (`debnce`). A typical debounce value is 25 ms.†
- TX watermark field (`tx_wmark`) of the FIFO threshold watermark register (`fifoth`). Typically, the threshold value is set to 512, which is half the FIFO buffer depth.†
- RX watermark field (`rx_wmark`) of the `fifoth` register. Typically, the threshold value is set to 511.†

These registers do not need to be changed with every SD/MMC/CE-ATA command. Set them to a typical value according to the SD/MMC/CE-ATA specifications.

Related Information

- **Clock Setup** on page 14-46
  Refer to this section for information on setting the clock source assignments.
- **Enumerated Card Stack** on page 14-43
  Refer to this section for information on discovering the card stack according to the card type.

Enumerated Card Stack

The card stack performs the following tasks:

- Discovers the connected card †
- Sets the relative Card Address Register (RCA) in the connected card†
- Reads the card specific information†
- Stores the card specific information locally†

The card connected to the controller can be an MMC, CE-ATA, SD or SDIO (including IO ONLY, MEM ONLY and COMBO) card.

Identifying the Connected Card Type

To identify the connected card type, the following discovery sequence is needed:

1. Reset the card width 1 or 4 bit (`card_width2`) and card width 8 bit (`card_width1`) fields in the `ctype` register to 0.
2. Identify the card type as SD, MMC, SDIO or SDIO-COMBO:

a. Send an SD/SDIO IO_SEND_OP_COND (CMD5) command with argument 0 to the card.
b. Read resp0 on the controller. The response to the IO_SEND_OP_COND command gives the voltage that the card supports.
c. Send the IO_SEND_OP_COND command, with the desired voltage window in the arguments. This command sets the voltage window and makes the card exit the initialization state.
d. Check bit 27 in resp0:

- If bit 27 is 0, the SDIO card is IO ONLY. In this case, proceed to **step 5**.
- If bit 27 is 1, the card type is SDIO COMBO. Continue with the following steps.

3. Go to **Card Type is Either SDIO COMBO or Still in Initialization** on page 14-44.
4. Go to **Determine if Card is a CE-ATA 1.1, CE-ATA 1.0, or MMC Device** on page 14-45.
5. At this point, the software has determined the card type as SD/SDHC, SDIO or SDIO-COMBO. Now it must enumerate the card stack according to the type that has been discovered.
6. Set the card clock source frequency to the frequency of identification clock rate, 400 KHz. Use one of the following discovery command sequences:

- For an SD card or an SDIO memory section, send the following SD/SDIO command sequence:
  - GO_IDLE_STATE
  - SEND_IF_COND
  - SD_SEND_OP_COND (ACMD41)
  - ALL_SEND_CID (CMD2)
  - SEND_RELATIVE_ADDR (CMD3)
- For an SDIO card, send the following command sequence:
  - IO_SEND_OP_COND
  - If the function count is valid, send the SEND_RELATIVE_ADDR command.
- For an MMC, send the following command sequence:
  - GO_IDLE_STATE
  - SEND_OP_COND (CMD1)
  - ALL_SEND_CID
  - SEND_RELATIVE_ADDR

7. You can change the card clock frequency after discovery by writing a value to the clkdiv register that divides down the sdmmc_clk clock.

   The following list shows typical clock frequencies for various types of cards:

- SD memory card, 25 MHz†
- MMC card device, 12.5 MHz†
- Full speed SDIO, 25 MHz†
- Low speed SDIO, 400 kHz†

Related Information

SD Association

To learn more about how SD technology works, visit the SD Association website (www.sdcard.org).

Card Type is Either SDIO COMBO or Still in Initialization

Only continue with this step if the SDIO card type is COMBO or there is no response received from the previous IO_SEND_OP_COND command. Otherwise, skip to **step 5** of the *Identifying the Connected Card Type* section.

1. Send the SD/SDIO SEND_IF_COND (CMD8) command with the following arguments:

- Bit[31:12] = 0x0 (reserved bits)†
- Bit[11:8] = 0x1 (supply voltage value)†
- Bit[7:0] = 0xAA (preferred check pattern by SD memory cards compliant with SDIO Simplified Specification Version 2.00 and later.)†

Refer to *SDIO Simplified Specification Version 2.00* as described on the SD Association website.

- If a response is received to the previous SEND_IF_COND command, the card supports SD High-Capacity, compliant with *SD Specifications, Part 1, Physical Layer Simplified Specification Version 2.00*.
- If no response is received, proceed to **the next decision statement**.

2. Send the SD_SEND_OP_COND (ACMD41) command with the following arguments:

- Bit[31] = 0x0 (reserved bits)†
- Bit[30] = 0x1 (high capacity status)†
- Bit[29:25] = 0x0 (reserved bits)†
- Bit[24] = 0x1 (S18R --supports voltage switching for 1.8V)†
- Bit[23:0] = supported voltage range†

- If the previous SD_SEND_OP_COND command receives a response, then the card type is SDHC. Otherwise, the card is MMC or CE-ATA. In either case, skip the following steps and proceed to **step 5** of the "Identifying the Connected Card Type" section.
- If the initial SEND_IF_COND command does not receive a response, then the card does not support High Capacity SD2.0. Now, proceed to **step step 3**.

3. Next, issue the GO_IDLE_STATE command followed by the SD_SEND_OP_COND command with the following arguments:

- Bit[31] = 0x0 (reserved bits)†
- Bit[30] = 0x0 (high capacity status)†
- Bit[29:24] = 0x0 (reserved bits)†
- Bit[23:0] = supported voltage range†

If a response is received to the previous SD_SEND_OP_COND command, the card is SD type. Otherwise, the card is MMC or CE-ATA.

Note: You must issue the SEND_IF_COND command prior to the first SD_SEND_OP_COND command, to initialize the High Capacity SD memory card. The card returns busy as a response to the SD_SEND_OP_COND command when any of the following conditions are true:

- The card executes its internal initialization process.
- A SEND_IF_COND command is not issued before the SD_SEND_OP_COND command.
- The ACMD41 command is issued. In the command argument, the Host Capacity Support (HCS) bit is set to 0, for a high capacity SD card.

Determine if Card is a CE-ATA 1.1, CE-ATA 1.0, or MMC Device

Use the following sequence to determine whether the card is a CE-ATA 1.1, CE-ATA 1.0, or MMC device:

Determine whether the card is a CE-ATA v1.1 card device by attempting to select ATA mode.

1. Send the SD/SDIO SEND_IF_COND command, querying byte 504 (S_CMD_SET) of the EXT_CSD register block in the external card.

If bit 4 is set to 1, the card device supports ATA mode.

2. Send the `SWITCH_FUNC (CMD6)` command, setting the ATA bit (bit 4) of the EXT_CSD register slice 191 (CMD_SET) to 1.

   This command selects ATA mode and activates the ATA command set.
3. You can verify the currently selected mode by reading it back from byte 191 of the EXT_CSD register.
4. Skip to **step 5** of the *Identifying the Connected Card Type* section.

   If the card device does not support ATA mode, it might be an MMC card or a CE-ATA v1.0 card. Proceed to the **next section** to determine whether the card is a CE-ATA 1.0 card device or an MMC card device.

Determine whether the card is a CE-ATA 1.0 card device or an MMC card device by sending the RW_REG command.

If a response is received and the response data contains the CE-ATA signature, the card is a CE-ATA 1.0 card device. Otherwise, the card is an MMC card device.

Clock Setup

The following registers of the SD/MMC controller allow software to select the desired clock frequency for the card:

- `clksrc`
- `clkdiv`
- `clkena`

The controller loads these registers when it receives an update clocks command.

Changing the Card Clock Frequency

To change the card clock frequency, perform the following steps:

1. Before disabling the clocks, ensure that the card is not busy with any previous data command. To do so, verify that the `data_busy` bit of the status register (`status`) is 0.
2. Reset the `cclk_enable` bit of the `clkena` register to 0, to disable the card clock generation.
3. Reset the `clksrc` register to 0.
4. Set the following bits in the `cmd` register to 1:
   - `update_clk_regs_only`—Specifies the update clocks command†
   - `wait_prvdata_complete`—Ensures that clock parameters do not change until any ongoing data transfer is complete†
   - `start_cmd`—Initiates the command†
5. Wait until the `start_cmd` and `update_clk_regs_only` bits change to 0. There is no interrupt when the clock modification completes. The controller does not set the `command_done` bit in the `rintsts` register upon command completion. The controller might signal a hardware lock error if it already has another command in the queue. In this case, return to **Step 4**.

   For information about hardware lock errors, refer to the "Interrupt and Error Handling" chapter.
6. Reset the `sdmmc_clk_enable` bit to 0 in the `enable` register of the clock manager peripheral PLL group (`perpllgrp`).
7. In the control register (`ctrl`) of the SDMMC controller group (`sdmmcgrp`) in the system manager, set the drive clock phase shift select (`drvsel`) and sample clock phase shift select (`smplsel`) bits to specify the required phase shift value.
8. Set the `sdmmc_clk_enable` bit in the `Enable` register of the clock manager `perpllgrp` group to 1.
9. Set the `clkdiv` register of the controller to the correct divider value for the required clock frequency.
10. Set the `cclk_enable` bit of the `clkena` register to 1, to enable the card clock generation.

You can also use the clkena register to enable low-power mode, which automatically stops the sdmmc_cclk_out clock when the card is idle for more than eight clock cycles.

Related Information

Interrupt and Error Handling on page 14-72
Refer to this section for information about hardware lock errors.

Timing Tuning

This section is pending further information.

Controller/DMA/FIFO Buffer Reset Usage

The following list shows the effect of reset on various parts in the SD/MMC controller:†

- Controller reset—resets the controller by setting the controller_reset bit in the ctrl register to 1. Controller reset resets the CIU and state machines, and also resets the BIU-to-CIU interface. Because this reset bit is self-clearing, after issuing the reset, wait until this bit changes to 0.†
- FIFO buffer reset—resets the FIFO buffer by setting the FIFO reset bit (fifo_reset) in the ctrl register to 1. FIFO buffer reset resets the FIFO buffer pointers and counters in the FIFO buffer. Because this reset bit is self-clearing, after issuing the reset, wait until this bit changes to 0.†
- DMA reset—resets the internal DMA controller logic by setting the DMA reset bit (dma_reset) in the ctrl register to 1, which immediately terminates any DMA transfer in progress. Because this reset bit is self-clearing, after issuing the reset, wait until this bit changes to 0.†

Note: Ensure that the DMA is idle before performing a DMA reset. Otherwise, the L3 interconnect might be left in an indeterminate state.†

Altera recommends setting the controller_reset, fifo_reset, and dma_reset bits in the ctrl register to 1 first, and then resetting the rintsts register to 0 using another write, to clear any resultant interrupt.

Non-Data Transfer Commands

To send any non-data transfer command, the software needs to write the cmd register and the cmdarg register with appropriate parameters. Using these two registers, the controller forms the command and sends it to the CMD pin. The controller reports errors in the command response through the error bits of the rintsts register.†

When a response is received—either erroneous or valid—the controller sets the command_done bit in the rintsts register to 1. A short response is copied to resp0, while a long response is copied to all four response registers (resp0, resp1, resp2, and resp3).† For long responses, bit 31 of resp3 represents the MSB and bit 0 of resp0 represents the LSB.†

For basic and non-data transfer commands, perform the following steps:

1. Write the cmdarg register with the appropriate command argument parameter.†
2. Write the cmd register with the settings in *Register Settings for Non-Data Transfer Command*.†
3. Wait for the controller to accept the command. The start_cmd bit changes to 0 when the command is accepted.†

   The following actions occur when the command is loaded into the controller:†

- If no previous command is being processed, the controller accepts the command for execution and resets the `start_cmd` bit in the `cmd` register to 0. If a previous command is being processed, the controller loads the new command in the command buffer.†
- If the controller is unable to load the new command—that is, a command is already in progress, a second command is in the buffer, and a third command is attempted—the controller generates a hardware lock error.†

4. Check if there is a hardware lock error.†
5. Wait for command execution to complete. After receiving either a response from a card or response timeout, the controller sets the `command_done` bit in the `rintsts` register to 1. Software can either poll for this bit or respond to a generated interrupt (if enabled).†
6. Check if the response timeout boot acknowledge received (`bar`), `rcrc`, or `re` bit is set to 1. Software can either respond to an interrupt raised by these errors or poll the `re`, `rcrc`, and `bar` bits of the `rintsts` register. If no response error is received, the response is valid. If required, software can copy the response from the response registers.†

Note: Software cannot modify clock parameters while a command is being executed.†

Related Information

cmd Register Settings for Non-Data Transfer Command† on page 14-48
Refer to this table for information about Non-Data Transfer commands.

cmd Register Settings for Non-Data Transfer Command†

Table 14-20: Default

Parameter	Value	Comment
`start_cmd`	1	This bit resets itself to 0 after the command is committed.
`use_hold_reg`	1 or 0	Choose the value based on the speed mode used.
`update_clk_regs_only`	0	Indicates that the command is not a clock update command
`data_expected`	0	Indicates that the command is not a data command
`card_number`	1	For one card
`cmd_index`	Command Index	Set this parameter to the command number. For example, set to 8 for the SD/SDIO SEND_IF_COND (CMD8) command.
`send_initialization`	0 or 1	1 for card reset commands such as the SD/SDIO GO_IDLE_STATE command 0 otherwise

Parameter	Value	Comment
stop_abort_cmd	0 or 1	1 for a command to stop data transfer, such as the SD/SDIO STOP_TRANSMISSION command 0 otherwise
response_length	0 or 1	1 for R2 (long) response 0 for short response
response_expect	0 or 1	0 for commands with no response, such as SD/SDIO GO_IDLE_STATE, SET_DSR (CMD4), or GO_INACTIVE_STATE (CMD15). 1 otherwise

Table 14-21: User Selectable

Parameter	Value	Comment
wait_prvdata_complete	1	Before sending a command on the command line, the host must wait for completion of any data command already in process. Altera recommends that you set this bit to 1, unless the current command is to query status or stop data transfer when transfer is in progress.
check_response_crc	1 or 0	1 if the response includes a valid CRC, and the software is required to crosscheck the response CRC bits. 0 otherwise

Data Transfer Commands

Data transfer commands transfer data between the memory card and the controller. To issue a data command, the controller requires a command argument, total data size, and block size. Data transferred to or from the memory card is buffered by the controller FIFO buffer.†

Confirming Transfer State

Before issuing a data transfer command, software must confirm that the card is not busy and is in a transfer state, by performing the following steps:†

1. Issue an SD/SDIO SEND_STATUS (CMD13) command. The controller sends the status of the card as the response to the command.†
2. Check the card's busy status.†
3. Wait until the card is not busy.†
4. Check the card's transfer status. If the card is in the stand-by state, issue an SD/SDIO SELECT/DESELECT_CARD (CMD7) command to place it in the transfer state.†

Busy Signal After CE-ATA RW_BLK Write Transfer

During CE-ATA RW_BLK write transfers, the MMC busy signal might be asserted after the last block. If the CE-ATA card device interrupt is disabled (the nIEN bit in the card device's ATA control register is set to 1), the `dto` bit in the `rintsts` register is set to 1 even though the card sends MMC BUSY. The host cannot issue the CMD60 command to check the ATA busy status after a CMD61 command. Instead, the host must perform one of the following actions:†

- Issue the SEND_STATUS command and check the MMC busy status before issuing a new CMD60 command†
- Issue the CMD39 command and check the ATA busy status before issuing a new CMD60 command†

For the data transfer commands, software must set the `ctype` register to the bus width that is programmed in the card.†

Data Transfer Interrupts

The controller generates an interrupt for different conditions during data transfer, which are reflected in the following `rintsts` register bits:†

1. `dto`—Data transfer is over or terminated. If there is a response timeout error, the controller does not attempt any data transfer and the Data Transfer Over bit is never set.†
2. Transmit FIFO data request bit (`txdr`)—The FIFO buffer threshold for transmitting data is reached; software is expected to write data, if available, into the FIFO buffer.†
3. Receive FIFO data request bit (`rxdr`)—The FIFO buffer threshold for receiving data is reached; software is expected to read data from the FIFO buffer.†
4. `hto`—The FIFO buffer is empty during transmission or is full during reception. Unless software corrects this condition by writing data for empty condition, or reading data for full condition, the controller cannot continue with data transfer. The clock to the card is stopped.†
5. `bds`—The card has not sent data within the timeout period.†
6. `dcrc`—A CRC error occurred during data reception.†
7. `sbe`—The start bit is not received during data reception.†
8. `ebe`—The end bit is not received during data reception, or for a write operation. A CRC error is indicated by the card.†

`dcrc`, `sbe`, and `ebe` indicate that the received data might have errors. If there is a response timeout, no data transfer occurs.†

Single-Block or Multiple-Block Read

To implement a single-block or multiple-block read, the software performs the following steps:†

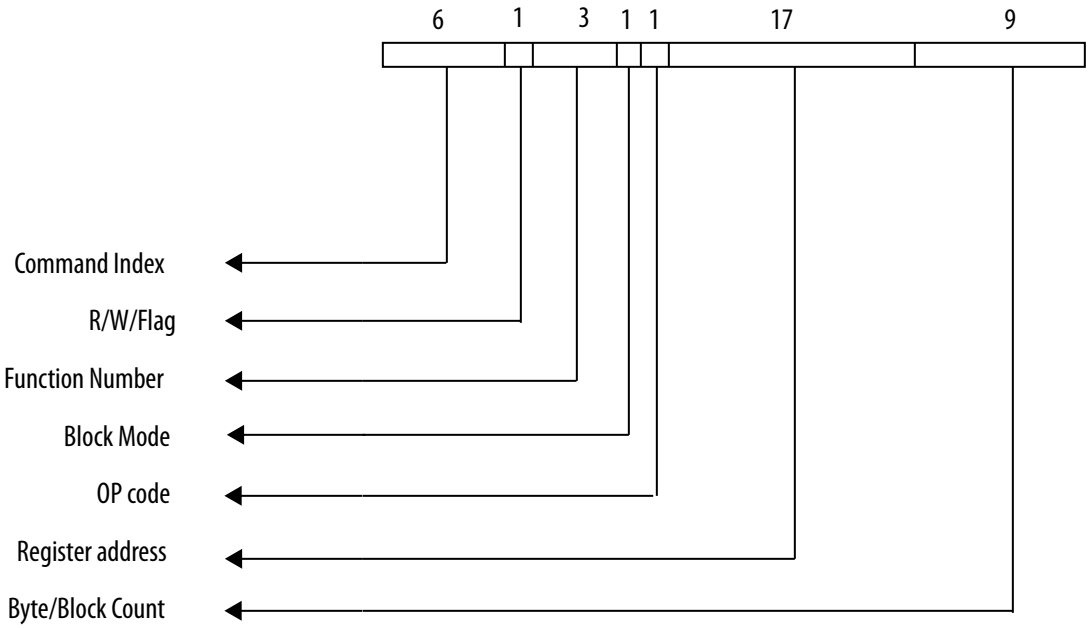
1. Write the data size in bytes to the `bytcnt` register. For a multi-block read, `bytcnt` must be a multiple of the block size.†
2. Write the block size in bytes to the `blksiz` register. The controller expects data to return from the card in blocks of size `blksiz`.†
3. If the read round trip delay, including the card delay, is greater than half of `sdmmc_clk_divided`, write to the card threshold control register (`cardthrctl`) to ensure that the card clock does not stop in the middle of a block of data being transferred from the card to the host. For more information, refer to *Card Read Threshold*.†

   Note: If the card read threshold enable bit (`cardrdthren`) is 0, the host system must ensure that the RX FIFO buffer does not become full during a read data transfer by ensuring that the RX FIFO buffer is read at a rate faster than that at which data is written into the FIFO buffer. Otherwise, an overflow might occur.†
4. Write the `cmdarg` register with the beginning data address for the data read.†

5. Write the `cmd` register with the parameters listed in *cmd Register Settings for Single-Block and Multiple-Block Reads*. For SD and MMC cards, use the SD/SDIO READ_SINGLE_BLOCK (CMD17) command for a single-block read and the READ_MULTIPLE_BLOCK (CMD18) command for a multiple-block read. For SDIO cards, use the IO_RW_EXTENDED (CMD53) command for both single-block and multiple-block transfers. The command argument for (CMD53) is shown in the figure, below. After writing to the `cmd` register, the controller starts executing the command. When the command is sent to the bus, the Command Done interrupt is generated.†
6. Software must check for data error interrupts, reported in the `dcrc`, `bds`, `sbe`, and `ebe` bits of the `rintsts` register. If required, software can terminate the data transfer by sending an SD/SDIO STOP command.†
7. Software must check for host timeout conditions in the `rintsts` register:†
   - Receive FIFO buffer data request†
   - Data starvation from host—the host is not reading from the FIFO buffer fast enough to keep up with data from the card. To correct this condition, software must perform the following steps:†
     - Read the `fifo_count` field of the `status` register†
     - Read the corresponding amount of data out of the FIFO buffer†

   In both cases, the software must read data from the FIFO buffer and make space in the FIFO buffer for receiving more data.†
8. When a DTO interrupt is received, the software must read the remaining data from the FIFO buffer.†

Figure 14-15: Command Argument for IO_RW_EXTENDED (CMD53)†



Related Information

- **Card Read Threshold** on page 14-69
  Refer to this section for information about the thresholds for a card read.

- **cmd Register Settings for Single-Block and Multiple-Block Reads†** on page 14-52
  Refer to this table for information about the settings for Single-Block and Multiple-Block Reads.

cmd Register Settings for Single-Block and Multiple-Block Reads†

Table 14-22: cmd Register Settings for Single-Block and Multiple-Block Reads (Default)

Parameter	Value	Comment
start_cmd	1	This bit resets itself to 0 after the command is committed.
use_hold_reg	1 or 0	Choose the value based on speed mode used.
update_clk_regs_only	0	Does not need to update clock parameters
data_expected	1	Data command
card_number	1	For one card
transfer_mode	0	Block transfer
send_initialization	0	1 for a card reset command such as the SD/SDIO GO_IDLE_STATE command 0 otherwise
stop_abort_cmd	0	1 for a command to stop data transfer such as the SD/SDIO STOP_TRANSMISSION command 0 otherwise
send_auto_stop	0 or 1	Refer to *Auto Stop* for information about how to set this parameter.
read_write	0	Read from card
response_length	0	1 for R2 (long) response 0 for short response
response_expect	1 or 0	0 for commands with no response, such as SD/SDIO GO_IDLE_STATE, SET_DSR, and GO_INACTIVE_STATE. 1 otherwise

Table 14-23: cmd Register Settings for Single-Block and Multiple-Block Reads (User Selectable)

Parameter	Value	Comment
wait_prvdata_complete	1 or 0	0 - sends command to CIU immediately 1 - sends command after previous data transfer ends

Parameter	Value	Comment
check_response_crc	1 or 0	0 - Controller must not check response CRC 1 - Controller must check responce CRC
cmd_index	Command Index	Set this parameter to the command number. For example, set to 17 or 18 for SD/SDIO READ_SINGLE_BLOCK (CMS17) or READ_MULTIPLE_BLOCK (CMD18)

Related Information

Auto-Stop on page 14-29
Refer to this table for information about setting the send_auto_stop parameter.

Single-Block or Multiple-Block Write

The following steps comprise a single-block or multiple-block write:

1. Write the data size in bytes to the bytcnt register. For a multi-block write, bytcnt must be a multiple of the block size.†
2. Write the block size in bytes to the blksiz register. The controller sends data in blocks of size blksiz each.†
3. Write the cmdarg register with the data address to which data must be written.†
4. Write data into the FIFO buffer. For best performance, the host software should write data continuously until the FIFO buffer is full.†
5. Write the cmd register with the parameters listed in *cmd Register Settings for Single-Block and Multiple-Block Write*. For SD and MMC cards, use the SD/SDIO WRITE_BLOCK (CMD24) command for a single-block write and the WRITE_MULTIPLE_BLOCK (CMD25) command for a multiple-block writes. For SDIO cards, use the IO_RW_EXTENDED command for both single-block and multiple-block transfers.†

   After writing to the cmd register, the controller starts executing a command if there is no other command already being processed. When the command is sent to the bus, a Command Done interrupt is generated.†
6. Software must check for data error interrupts; that is, for dcrc, bds, and ebe bits of the rintsts register. If required, software can terminate the data transfer early by sending the SD/SDIO STOP command.†
7. Software must check for host timeout conditions in the rintsts register: †
   - Transmit FIFO buffer data request.†
   - Data starvation by the host—the controller wrote data to the card faster than the host could supply the data.†

   In both cases, the software must write data into the FIFO buffer.†

   There are two types of transfers: open-ended and fixed length.†

- Open-ended transfers—For an open-ended block transfer, the byte count is 0. At the end of the data transfer, software must send the STOP_TRANSMISSION command (CMD12).†
- Fixed-length transfers—The byte count is nonzero. You must already have written the number of bytes to the `bytcnt` register. The controller issues the STOP command for you if you set the `send_auto_stop` bit of the `cmd` register to 1. After completion of a transfer of a given number of bytes, the controller sends the STOP command. Completion of the AUTO_STOP command is reflected by the Auto Command Done interrupt. A response to the AUTO_STOP command is written to the `resp1` register. If software does not set the `send_auto_stop` bit in the `cmd` register to 1, software must issue the STOP command just like in the open-ended case.†

When the `dto` bit of the `rintsts` register is set, the data command is complete.†

cmd Register Settings for Single-Block and Multiple-Block Write

Table 14-24: cmd Register Settings for Single-Block and Multiple-Block Write (Default)†

Parameter	Value	Comment
`start_cmd`	1	This bit resets itself to 0 after the command is committed (accepted by the BIU).
`use_hold_reg`	1 or 0	Choose the value based on speed mode used.
`update_clk_regs_only`	0	Does not need to update clock parameters
`data_expected`	1	Data command
`card_number`	1	For one card
`transfer_mode`	0	Block transfer
`send_initialization`	0	Can be 1, but only for card reset commands such as SD/SDIO GO_IDLE_STATE
`stop_abort_cmd`	0	Can be 1 for commands to stop data transfer such as SD/SDIO STOP_TRANSMISSION
`send_auto_stop`	0 or 1	Refer to *Auto Stop* for information about how to set this parameter.
`read_write`	1	Write to card
`response_length`	0	Can be 1 for R2 (long) responses
`response_expect`	1	Can be 0 for commands with no response. For example, SD/SDIO GO_IDLE_STATE, SET_DSR, GO_INACTIVE_STATE etc.

Table 14-25: cmd Register Settings for Single-Block and Multiple-Block Write (User Selectable)†

Parameter	Value	Comment
wait_prvdata_complete	1	0—Sends command to the CIU immediately 1—Sends command after previous data transfer ends
check_response_crc	1	0—Controller must not check response CRC 1—Controller must check response CRC
cmd_index	Command Index	Set this parameter to the command number. For example, set to 24 for SD/SDIO WRITE_BLOCK (CMD24) or 25 for WRITE_MULTIPLE_BLOCK (CMD25).

Related Information

Auto-Stop on page 14-29
Refer to this table for information about setting the send_auto_stop parameter.

Stream Read and Write

In a stream transfer, if the byte count is equal to 0, the software must also send the SD/SDIO STOP command. If the byte count is not 0, when a given number of bytes completes a transfer, the controller sends the STOP command automatically. Completion of this AUTO_STOP command is reflected by the Auto_command_done interrupt. A response to an AUTO_STOP command is written to the resp1 register. A stream transfer is allowed only for card interfaces with a 1-bit data bus.†

A stream read requires the same steps as the block read described in *Single-Block or Multiple-Block Read*, except for the following bits in the cmd register:†

- transfer_mode = 0x1 (for stream transfer)†
- cmd_index = 20 (SD/SDIO CMD20)†

A stream write requires the same steps as the block write mentioned in *Single-Block or Multiple-Block Write*, except for the following bits in the cmd register:†

- transfer_mode = 0x1 (for stream transfer)†
- cmd_index = 11 (SD/SDIO CMD11)†

Related Information

- **Single-Block or Multiple-Block Read** on page 14-50
  Refer to this section for more information about a stream read.
- **Single-Block or Multiple-Block Write** on page 14-53
  Refer to this section for more information about a stream write.

Packed Commands

To reduce overhead, read and write commands can be packed in groups of commands—either all read or all write—that transfer the data for all commands in the group in one transfer on the bus. Use the SD/SDIO SET_BLOCK_COUNT (CMD23) command to state ahead of time how many blocks will be transferred. Then issue a single READ_MULTIPLE_BLOCK or WRITE_MULTIPLE_BLOCK command to read or write multiple blocks.†

- SET_BLOCK_COUNT—set block count (number of blocks transferred using the READ_MULTIPLE_BLOCK or WRITE_MULTIPLE_BLOCK command) †
- READ_MULTIPLE_BLOCK—multiple-block read command †
- WRITE_MULTIPLE_BLOCK—multiple-block write command†

Packed commands are organized in packets by the application software and are transparent to the controller.†

Related Information

www.jedec.org
For more information about packed commands, refer to JEDEC Standard No. 84-A441, available on the JEDEC website.

Transfer Stop and Abort Commands

This section describes stop and abort commands. The SD/SDIO STOP_TRANSMISSION command can terminate a data transfer between a memory card and the controller. The ABORT command can terminate an I/O data transfer for only an SDIO card. †

STOP_TRANSMISSION (CMD12)

The host can send the STOP_TRANSMISSION (CMD12) command on the CMD pin at any time while a data transfer is in progress. Perform the following steps to send the STOP_TRANSMISSION command to the SD/SDIO card device:†

1. Set the `wait_prvdata_complete` bit of the `cmd` register to 0.†
2. Set the `stop_abort_cmd` in the `cmd` register to 1, which ensures that the CIU stops.†

The STOP_TRANSMISSION command is a non-data transfer command.†

Related Information

Non-Data Transfer Commands on page 14-47
Refer to this section for information on the STOP_TRANSMISSION command.

ABORT

The ABORT command can only be used with SDIO cards. To abort the function that is transferring data, program the ABORT function number in the ASx[2:0] bits at address 0x06 of the card common control register (CCCR) in the card device, using the IO_RW_DIRECT (CMD52) command. The CCCR is located at the base of the card space 0x00 – 0xFF.†

Note: The ABORT command is a non-data transfer command.†

Related Information

Non-Data Transfer Commands on page 14-47
Refer to this section for information on the ABORT command.

Sending the ABORT Command

Perform the following steps to send the ABORT command to the SDIO card device:†

1. Set the `cmdarg` register to include the appropriate command argument parameters listed in *cmdarg Register Settings for SD/SDIO ABORT Command*.†
2. Send the IO_RW_DIRECT command by setting the following fields of the `cmd` register:†

- Set the command index to 0x52 (IO_RW_DIRECT).†
- Set the `stop_abort_cmd` bit of the `cmd` register to 1 to inform the controller that the host aborted the data transfer.†
- Set the `wait_prvdata_complete` bit of the `cmd` register to 0.†

3. Wait for the `cmd` bit in the `rintsts` register to change to 1.†
4. Read the response to the IO_RW_DIRECT command (R5) in the response registers for any errors.†

For more information about response values, refer to the *Physical Layer Simplified Specification*, Version 3.01, available on the SD Association website.

Related Information

SD Association

To learn more about how SD technology works, visit the SD Association website (www.sdcard.org).

cmdarg Register Settings for SD/SDIO ABORT Command†

Table 14-26: cmdarg Register Settings for SD/SDIO ABORT Command

Bits	Contents	Value
31	R/W flag	1
30:28	Function number	0, for access to the CCCR in the card device
27	RAW flag	1, if needed to read after write
26	Don't care	-
25:9	Register address	0x06
8	Don't care	-
7:0	Write data	Function number to abort

Internal DMA Controller Operations

For better performance, you can use the internal DMA controller to transfer data between the host and the controller. This section describes the internal DMA controller's initialization process, and transmission sequence, and reception sequence.

Internal DMA Controller Initialization

To initialize the internal DMA controller, perform the following steps:†

1. Set the required `bmod` register bits: †

- If the internal DMA controller enable bit (de) of the bmod register is set to 0 during the middle of a DMA transfer, the change has no effect. Disabling only takes effect for a new data transfer command.†
- Issuing a software reset immediately terminates the transfer. Prior to issuing a software reset, Altera recommends the host reset the DMA interface by setting the dma_reset bit of the ctrl register to 1.†
- The pbl field of the bmod register is read-only and a direct reflection of the contents of the DMA multiple transaction size field (dw_dma_multiple_transaction_size) in the fifoth register.†
- The fb bit of the bmod register has to be set appropriately for system performance.†

2. Write to the idinten register to mask unnecessary interrupt causes according to the following guidelines:†
   - When a Descriptor Unavailable interrupt is asserted, the software needs to form the descriptor, appropriately set its own bit, and then write to the poll demand register (pldmnd) for the internal DMA controller to re-fetch the descriptor.†
   - It is always appropriate for the software to enable abnormal interrupts because any errors related to the transfer are reported to the software.†
3. Populate either a transmit or receive descriptor list in memory. Then write the base address of the first descriptor in the list to the internal DMA controller's descriptor list base address register (dbaddr). The DMA controller then proceeds to load the descriptor list from memory. *Internal DMA Controller Transmission Sequences* and *Internal DMA Controller Reception Sequences* describe this step in detail. †

Related Information

- **Internal DMA Controller Transmission Sequences** on page 14-58
  Refer to this section for information about the Internal DMA Controller Transmission Sequences.
- **Internal DMA Controller Reception Sequences** on page 14-59
  Refer to this section for information about the Internal DMA Controller Reception Sequences.

Internal DMA Controller Transmission Sequences

To use the internal DMA controller to transmit data, perform the following steps:

1. The host sets up the Descriptor fields (DES0—DES3) for transmission and sets the OWN bit (DES0[31]) to 1. The host also loads the data buffer in system memory with the data to be written to the SD card.†
2. The host writes the appropriate write data command (SD/SDIO WRITE_BLOCK or WRITE_MULTIPLE_BLOCK) to the cmd register. The internal DMA controller determines that a write data transfer needs to be performed.†
3. The host sets the required transmit threshold level in the tx_wmark field in the fifoth register.†
4. The internal DMA controller engine fetches the descriptor and checks the OWN bit. If the OWN bit is set to 0, the host owns the descriptor. In this case, the internal DMA controller enters the suspend state and asserts the Descriptor Unable interrupt. The host then needs to set the descriptor OWN bit to 1 and release the DMA controller by writing any value to the pldmnd register.†
5. The host must write the descriptor base address to the dbaddr register.†
6. The internal DMA controller waits for the Command Done (CD) bit in the rintsts register to be set to 1, with no errors from the BIU. This condition indicates that a transfer can be done.†
7. The internal DMA controller engine waits for a DMA interface request from BIU. The BIU divides each transfer into smaller chunks. Each chunk is an internal request to the DMA. This request is generated based on the transmit threshold value.†

8. The internal DMA controller fetches the transmit data from the data buffer in the system memory and transfers the data to the FIFO buffer in preparation for transmission to the card.†
9. When data spans across multiple descriptors, the internal DMA controller fetches the next descriptor and continues with its operation with the next descriptor. The Last Descriptor bit in the descriptor DES0 field indicates whether the data spans multiple descriptors or not.†
10. When data transmission is complete, status information is updated in the `idsts` register by setting the `ti` bit to 1, if enabled. Also, the OWN bit is set to 0 by the DMA controller by updating the DES0 field of the descriptor.†

Internal DMA Controller Reception Sequences

To use the internal DMA controller to receive data, perform the following steps:

1. The host sets up the descriptor fields (DES0—DES3) for reception and sets the OWN (DES0 [31]) to 1.†
2. The host writes the read data command to the `cmd` register in BIU. The internal DMA controller determines that a read data transfer needs to be performed.†
3. The host sets the required receive threshold level in the `rx_wmark` field in the `fifoth` register.†
4. The internal DMA controller engine fetches the descriptor and checks the OWN bit. If the OWN bit is set to 0, the host owns the descriptor. In this case, the internal DMA controller enters suspend state and asserts the Descriptor Unable interrupt. The host then must set the descriptor OWN bit to 1 and release the DMA controller by writing any value to the `pldmnd` register.†
5. The host must write the descriptor base address to the `dbaddr` register.†
6. The internal DMA controller waits for the `CD` bit in the `rintsts` register to be set to 1, with no errors from the BIU. This condition indicates that a transfer can be done.†
7. The internal DMA controller engine waits for a DMA interface request from the BIU. The BIU divides each transfer into smaller chunks. Each chunk is an internal request to the DMA. This request is generated based on the receive threshold value.†
8. The internal DMA controller fetches the data from the FIFO buffer and transfers the data to system memory.†
9. When data spans across multiple descriptors, the internal DMA controller fetches the next descriptor and continues with its operation with the next descriptor. The Last Descriptor bit in the descriptor indicates whether the data spans multiple descriptors or not.†
10. When data reception is complete, status information is updated in the `idsts` register by setting the `ri` bit to 1, if enabled. Also, the OWN bit is set to 0 by the DMA controller by updating the DES0 field of the descriptor.†

Commands for SDIO Card Devices

This section describes the commands to temporarily halt the transfers between the controller and SDIO card device.

Suspend and Resume Sequence

For SDIO cards, a data transfer between an I/O function and the controller can be temporarily halted using the SUSPEND command. This capability might be required to perform a high-priority data transfer with another function. When desired, the suspended data transfer can be resumed using the RESUME command.†

The SUSPEND and RESUME operations are implemented by writing to the appropriate bits in the CCCR (Function 0) of the SDIO card. To read from or write to the CCCR, use the controller's IO_RW_DIRECT command.†

Suspend

To suspend data transfer, perform the following steps:†

1. Check if the SDIO card supports the SUSPEND/RESUME protocol by reading the SBS bit in the CCCR at offset 0x08 of the card.†
2. Check if the data transfer for the required function number is in process. The function number that is currently active is reflected in the function select bits (FSx) of the CCCR, bits 3:0 at offset 0x0D of the card.†

   Note: If the bus status bit (BS), bit 0 at address 0xC, is 1, only the function number given by the FSx bits is valid.†
3. To suspend the transfer, set the bus release bit (BR), bit 2 at address 0xC, to 1.†
4. Poll the BR and BS bits of the CCCR at offset 0x0C of the card until they are set to 0. The BS bit is 1 when the currently-selected function is using the data bus. The BR bit remains 1 until the bus release is complete. When the BR and BS bits are 0, the data transfer from the selected function is suspended.†
5. During a read-data transfer, the controller can be waiting for the data from the card. If the data transfer is a read from a card, the controller must be informed after the successful completion of the SUSPEND command. The controller then resets the data state machine and comes out of the wait state. To accomplish this, set the abort read data bit (`abort_read_data`) in the `ctrl` register to 1.†
6. Wait for data completion, by polling until the `dto` bit is set to 1 in the `rintsts` register. To determine the number of pending bytes to transfer, read the transferred CIU card byte count (`tcbcnt`) register of the controller. Subtract this value from the total transfer size. You use this number to resume the transfer properly.†

Resume

To resume the data transfer, perform the following steps:†

1. Check that the card is not in a transfer state, which confirms that the bus is free for data transfer.†
2. If the card is in a disconnect state, select it using the SD/SDIO SELECT/DESELECT_CARD command. The card status can be retrieved in response to an IO_RW_DIRECT or IO_RW_EXTENDED command.†
3. Check that a function to be resumed is ready for data transfer. Determine this state by reading the corresponding RF<*n*> flag in CCCR at offset 0x0F of the card. If RF<*n*> = 1, the function is ready for data transfer.†

   Note: For detailed information about the RF<*n*> flags, refer to SDIO Simplified Specification Version 2.00, available on the SD Association website.†
4. To resume transfer, use the IO_RW_DIRECT command to write the function number at the FSx bits in the CCCR, bits 3:0 at offset 0x0D of the card. Form the command argument for the IO_RW_DIRECT command and write it to the `cmdarg` register. Bit values are listed in the following table.†

Table 14-27: cmdarg Bit Values for RESUME Command†

Bits	Content	Value
31	R/W flag	1
30:28	Function number	0, for CCCR access
27	RAW flag	1, read after write
26	Don't care	-

Bits	Content	Value
25:9	Register address	0x0D
8	Don't care	-
7:0	Write data	Function number that is to be resumed

5. Write the block size value to the `blksiz` register. Data is transferred in units of this block size.†
6. Write the byte count value to the `bytcnt` register. Specify the total size of the data that is the remaining bytes to be transferred. It is the responsibility of the software to handle the data.†

   To determine the number of pending bytes to transfer, read the transferred CIU card byte count register (`tcbcnt`). Subtract this value from the total transfer size to calculate the number of remaining bytes to transfer.†
7. Write to the `cmd` register similar to a block transfer operation. When the `cmd` register is written, the command is sent and the function resumes data transfer. For more information, refer to *Single-Block or Multiple-Block Read* and *Single-Block or Multiple-Block Write*.†
8. Read the resume data flag (DF) of the SDIO card device. Interpret the DF flag as follows:†
   - DF=1—The function has data for the transfer and begins a data transfer as soon as the function or memory is resumed.†
   - DF=0—The function has no data for the transfer. If the data transfer is a read, the controller waits for data. After the data timeout period, it issues a data timeout error.†

Related Information

- **SD Association**
  To learn more about how SD technology works, visit the SD Association website (www.sdcard.org).
- **Single-Block or Multiple-Block Read** on page 14-50
  Refer to this section for more information about writing to the `cmd` register.
- **Single-Block or Multiple-Block Write** on page 14-53
  Refer to this section for more information about writing to the `cmd` register.

Read-Wait Sequence

Read_wait is used with SDIO cards only. It temporarily stalls the data transfer, either from functions or memory, and allows the host to send commands to any function within the SDIO card device. The host can stall this transfer for as long as required. The controller provides the facility to signal this stall transfer to the card.†

Signalling a Stall

To signal the stall, perform the following steps:†

1. Check if the card supports the read_wait facility by reading the SDIO card's SRW bit, bit 2 at offset 0x8 in the CCCR.†
2. If this bit is 1, all functions in the card support the read_wait facility. Use the SD/SDIO IO_RW_DIRECT command to read this bit.†
3. If the card supports the `read_wait` signal, assert it by setting the read wait bit (`read_wait`) in the `ctrl` register to 1.†
4. Reset the `read_wait` bit to 0 in the `ctrl` register.†

CE-ATA Data Transfer Commands

This section describes CE-ATA data transfer commands.

Related Information

Data Transfer Commands on page 14-49
Refer to this section for information about the basic settings and interrupts generated for different conditions.

ATA Task File Transfer Overview

ATA task file registers are mapped to addresses 0x00h through 0x10h in the MMC register space. The RW_REG command is used to issue the ATA command, and the ATA task file is transmitted in a single RW_REG MMC command sequence.†

The host software stack must write the task file image to the FIFO buffer before setting the `cmdarg` and `cmd` registers in the controller. The host processor then writes the address and byte count to the `cmdarg` register before setting the `cmd` register bits.†

For the RW_REG command, there is no CCS from the CE-ATA card device. †

ATA Task File Transfer Using the RW_MULTIPLE_REGISTER (RW_REG) Command

This command involves data transfer between the CE-ATA card device and the controller. To send a data command, the controller needs a command argument, total data size, and block size. Software receives or sends data through the FIFO buffer.†

Implementing ATA Task File Transfer

To implement an ATA task file transfer (read or write), perform the following steps:†

1. Write the data size in bytes to the `bytcnt` register. `bytcnt` must equal the block size, because the controller expects a single block transfer.†
2. Write the block size in bytes to the `blksiz` register.†
3. Write the `cmdarg` register with the beginning register address.†

You must set the `cmdarg`, `cmd`, `blksiz`, and `bytcnt` registers according to the tables in *Register Settings for ATA Task File Transfer*.†

Related Information

Register Settings for ATA Task File Transfer on page 14-62
Refer to this table for information on how to set these registers.

Register Settings for ATA Task File Transfer

Table 14-28: cmdarg Register Settings for ATA Task File Transfer†

Bit	Value	Comment
31	1 or 0	Set to 0 for read operation or set to 1 for write operation
30:24	0	Reserved (bits set to 0 by host processor)
23:18	0	Starting register address for read or write (DWORD aligned)

Bit	Value	Comment
17:16	0	Register address (DWORD aligned)
15:8	0	Reserved (bits set to 0 by host processor)
7:2	16	Number of bytes to read or write (integral number of DWORD)
1:0	0	Byte count in integral number of DWORD

Table 14-29: cmd Register Settings for ATA Task File Transfer†

Bit	Value	Comment
`start_cmd`	1	
`ccs_expected`	0	CCS is not expected
`read_ceata_device`	0 or 1	Set to 1 if RW_BLK or RW_REG read
`update_clk_regs_only`	0	No clock parameters update command
`card_num`	0	
`send_initialization`	0	No initialization sequence
`stop_abort_cmd`	0	
`send_auto_stop`	0	
`transfer_mode`	0	Block transfer mode. Block size and byte count must match number of bytes to read or write
`read_write`	1 or 0	1 for write and 0 for read
`data_expected`	1	Data is expected
`response_length`	0	
`response_expect`	1	
`cmd_index`	Command index	Set this parameter to the command number. For example, set to 24 for SD/SDIO WRITE_BLOCK (CMD24) or 25 for WRITE_MULTIPLE_BLOCK (CMD25).
`wait_prvdata_complete`	1	• 0 for send command immediately • 1 for send command after previous DTO interrupt

Bit	Value	Comment
check_response_crc	1	• 0 for not checking response CRC • 1 for checking response CRC

Table 14-30: blksiz Register Settings for ATA Task File Transfer†

Bit	Value	Comment
31:16	0	Reserved bits set to 0
15:0 (block_size)	16	For accessing entire task file (16, 8-bit registers). Block size of 16 bytes

Table 14-31: bytcnt Register Settings for ATA Task File Transfer

Bit	Value	Comment
31:0	16	For accessing entire task file (16, 8-bit registers). Byte count value of 16 is used with the block size set to 16.

Reset and Card Device Discovery Overview

Before starting any CE-ATA operations, the host must perform a MMC reset and initialization procedure. The host and card device must negotiate the MMC transfer (MMC TRAN) state before the card enters the MMC TRAN state.†

The host must follow the existing MMC discovery procedure to negotiate the MMC TRAN state. After completing normal MMC reset and initialization procedures, the host must query the initial ATA task file values using the RW_REG or CMD39 command.†

By default, the MMC block size is 512 bytes—indicated by bits 1:0 of the srcControl register inside the CE-ATA card device. The host can negotiate the use of a 1 KB or 4 KB MMC block sizes. The card indicates MMC block sizes that it can support through the srcCapabilities register in the MMC; the host reads this register to negotiate the MMC block size. Negotiation is complete when the host controller writes the MMC block size into the srcControl register bits 1:0 of the card.†

Related Information

www.jedec.org

For information about the (MMC TRAN) state, MMC reset and initialization, refer to JEDEC Standard No. 84-A441, available on the JEDEC website.

ATA Payload Transfer Using the RW_MULTIPLE_BLOCK (RW_BLK) Command

This command involves data transfer between the CE-ATA card device and the controller. To send a data command, the controller needs a command argument, total data size, and block size. Software receives or sends data through the FIFO buffer. †

Implementing ATA Payload Transfer

To implement an ATA payload transfer (read or write), perform the following steps:†

1. Write the data size in bytes to the `bytcnt` register.†
2. Write the block size in bytes to the `blksiz` register. The controller expects a single/multiple block transfer.†
3. Write to the `cmdarg` register to indicate the data unit count.†

Register Settings for ATA Payload Transfer

You must set the `cmdarg`, `cmd`, `blksiz`, and `bytcnt` registers according to the following tables.†

Table 14-32: cmdarg Register Settings for ATA Payload Transfer†

Bits	Value	Comment
31	1 or 0	Set to 0 for read operation or set to 1 for write operation
30:24	0	Reserved (bits set to 0 by host processor)
23:16	0	Reserved (bits set to 0 by host processor)
15:8	Data count	Data Count Unit [15:8]
7:0	Data count	Data Count Unit [7:0]

Table 14-33: cmd Register Settings for ATA Payload Transfer†

Bits	Value	Comment
`start_cmd`	1	-
`ccs_expected`	1	CCS is expected. Set to 1 for the RW_BLK command if interrupts are enabled in CE-ATA card device (the `nIEN` bit is set to 0 in the ATA control register)
`read_ceata_device`	0 or 1	Set to 1 for a RW_BLK or RW_REG read command
`update_clk_regs_only`	0	No clock parameters update command
`card_num`	0	-
`send_initialization`	0	No initialization sequence
`stop_abort_cmd`	0	-
`send_auto_stop`	0	-
`transfer_mode`	0	Block transfer mode. Byte count must be integer multiple of 4kB. Block size can be 512, 1k or 4k bytes

Bits	Value	Comment
read_write	1 or 0	1 for write and 0 for read
data_expected	1	Data is expected
response_length	0	-
response_expect	1	-
cmd_index	Command index	Set this parameter to the command number. For example, set to 24 for SD/SDIO WRITE_BLOCK (CMD24) or 25 for WRITE_MULTIPLE_BLOCK (CMD25).
wait_prvdata_complete	1	• 0 for send command immediately • 1 for send command after previous DTO interrupt
check_response_crc	1	• 0 for not checking response CRC • 1 for checking response CRC

Table 14-34: blksiz Register Settings for ATA Payload Transfer†

Bits	Value	Comment
31:16	0	Reserved bits set to 0
15:0 (block_size)	512, 1024 or 4096	MMC block size can be 512, 1024 or 4096 bytes as negotiated by host

Table 14-35: bytcnt Register Settings for ATA Payload Transfer

Bits	Value	Comment
31:0	<n>*block_size	Byte count must be an integer multiple of the block size. For ATA media access commands, byte count must be a multiple of 4 KB. (<*n*>*block_size = <*x*>*4 KB, where <*n*> and <*x*> are integers)

CE-ATA CCS

This section describes disabling the CCS, recovery after CCS timeout, and recovery after I/O read transmission delay (N_{ACIO}) timeout. †

Disabling the CCS

While waiting for the CCS for an outstanding RW_BLK command, the host can disable the CCS by sending a CCSD command:†

- Send a CCSD command—the controller sends the CCSD command to the CE-ATA card device if the `send_ccsd` bit is set to 1 in the `ctrl` register of the controller. This bit can be set only after a response is received for the RW_BLK command.†
- Send an internal stop command—send an internally-generated SD/SDIO STOP_TRANSMISSION (CMD12) command after sending the CCSD pattern. If the `send_auto_stop_ccsd` bit of the `ctrl` register is also set to 1 when the controller is set to send the CCSD pattern, the controller sends the internally-generated STOP command to the `CMD` pin. After sending the STOP command, the controller sets the `acd` bit in the `rintsts` register to 1.†

Recovery after CCS Timeout

If a timeout occurs while waiting for the CCS, the host needs to send the CCSD command followed by a STOP command to abort the pending ATA command. The host can set up the controller to send an internally-generated STOP command after sending the CCSD pattern:†

- Send CCSD command—set the `send_ccsd` bit in the `ctrl` register to 1.†
- Send external STOP command—terminate the data transfer between the CE-ATA card device and the controller. For more information about sending the STOP command, refer to *Transfer Stop and Abort Commands*.†
- Send internal STOP command—set the `send_auto_stop_ccsd` bit in the `ctrl` register to 1, which tells the controller to send the internally-generated STOP command. After sending the STOP command, the controller sets the `acd` bit in the `rintsts` register to 1. The `send_auto_stop_ccsd` bit must be set to 1 along with setting the `send_ccsd` bit.†

Related Information

Transfer Stop and Abort Commands on page 14-56
Refer to this section for more information about sending the STOP command.

Recovery after I/O Read Transmission Delay (N_{ACIO}) Timeout

If the I/O read transmission delay (N_{ACIO}) timeout occurs for the CE-ATA card device, perform one of the following steps to recover from the timeout:†

- If the CCS is expected from the CE-ATA card device (that is, the `ccs_expected` bit is set to 1 in the `cmd` register), follow the steps in *Recovery after CCS Timeout*.†
- If the CCS is not expected from the CE-ATA card device, perform the following steps: †
  1. Send an external STOP command. †
  2. Terminate the data transfer between the controller and CE-ATA card device. †

Related Information

Recovery after CCS Timeout on page 14-67
For more information about what steps to take if the CCS is expected from the CE-ATA card device.

Reduced ATA Command Set

It is necessary for the CE-ATA card device to support the reduced ATA command subset. This section describes the reduced command set.†

The IDENTIFY DEVICE Command

The IDENTIFY DEVICE command returns a 512-byte data structure to the host that describes device-specific information and capabilities. The host issues the IDENTIFY DEVICE command only if the MMC block size is set to 512 bytes. Any other MMC block size has indeterminate results.†

The host issues a RW_REG command for the ATA command, and the data is retrieved with the RW_BLK command.†

The host controller uses the following settings while sending a RW_REG command for the IDENTIFY DEVICE ATA command. The following list shows the primary bit settings:†

- `cmd` register setting: `data_expected` bit set to 0†
- `cmdarg` register settings: †
  - Bit [31] set to 0†
  - Bits [7:2] set to 128 †
  - All other bits set to 0†
- Task file settings: †
  - Command field of the ATA task file set to 0xEC†
  - Reserved fields of the task file set to 0†
- `bytcnt` register and `block_size` field of the `blksiz` register: set to 16†

The host controller uses the following settings for data retrieval (RW_BLK command):†

- `cmd` register settings:†
  - `ccs_expected` set to 1†
  - `data_expected` set to 1†
- `cmdarg` register settings: †
  - Bit [31] set to 0 (read operation) †
  - Bits [15:0] set to 1 (data unit count = 1)†
  - All other bits set to 0†
- `bytcnt` register and `block_size` field of the `blksiz` register: set to 512†

The READ DMA EXT Command

The READ DMA EXT command reads a number of logical blocks of data from the card device using the Data-In data transfer protocol. The host uses a RW_REG command to issue the ATA command and the RW_BLK command for the data transfer.†

The WRITE DMA EXT Command

The WRITE DMA EXT command writes a number of logical blocks of data to the card device using the Data-Out data transfer protocol. The host uses a RW_REG command to issue the ATA command and the RW_BLK command for the data transfer.†

The STANDBY IMMEDIATE Command

This ATA command causes the card device to immediately enter the most aggressive power management mode that still retains internal device context. No data transfer (RW_BLK) is expected for this command.†

For card devices that do not provide a power savings mode, the STANDBY IMMEDIATE command returns a successful status indication. The host issues a RW_REG command for the ATA command, and the status is retrieved with the SD/SDIO CMD39 or RW_REG command. Only the status field of the ATA task file contains the success status; there is no error status.†

The host controller uses the following settings while sending the RW_REG command for the STANDBY IMMEDIATE ATA command: †

- `cmd` register setting: `data_expected` bit set to 0†
- `cmdarg` register settings: †
  - Bit [31] set to 1 †
  - Bits [7:2] set to 4 †
  - All other bits set to 0 †
- Task file settings: †
  - Command field of the ATA task file set to 0xE0†
  - Reserved fields of the task file set to 0†
- `bytcnt` register and `block_size` field of the `blksiz` register: set to 16 †

The FLUSH CACHE EXT Command

For card devices that buffer/cache written data, the FLUSH CACHE EXT command ensures that buffered data is written to the card media. For cards that do not buffer written data, the FLUSH CACHE EXT command returns a success status. No data transfer (RW_BLK) is expected for this ATA command. †

The host issues a RW_REG command for the ATA command, and the status is retrieved with the SD/SDIO CMD39 or RW_REG command. There can be error status for this ATA command, in which case fields other than the status field of the ATA task file are valid.†

The host controller uses the following settings while sending the RW_REG command for the STANDBY IMMEDIATE ATA command:†

- `cmd` register setting: `data_expected` bit set to 0 †
- `cmdarg` register settings: †
  - Bit [31] set to 1 †
  - Bits [7:2] set to 4†
  - All other bits set to 0†
- Task file settings: †
  - Command field of the ATA task file set to 0xEA †
  - Reserved fields of the task file set to 0†
- `bytcnt` register and `block_size` field of the `blksiz` register: set to 16 †

Card Read Threshold

When an application needs to perform a single or multiple block read command, the application must set the `cardthrctl` register with the appropriate card read threshold size in the card read threshold field (`cardrdthreshold`) and set the `cardrdthren` bit to 1. This additional information specified in the controller ensures that the controller sends a read command only if there is space equal to the card read threshold available in the RX FIFO buffer. This in turn ensures that the card clock is not stopped in the middle a block of data being transmitted from the card. Set the card read threshold to the block size of the transfer to guarantee there is a minimum of one block size of space in the RX FIFO buffer before the controller enables the card clock. †

The card read threshold is required when the round trip delay is greater than half of `sdmmc_clk_divided`.†

Table 14-36: Card Read Threshold Guidelines†

Bus Speed Modes	Round Trip Delay (Delay_R) (43)	Is Stopping of Card Clock Allowed?	Card Read Threshold Required?
SDR25	Delay_R > 0.5 * (sdmmc_clk/4)	No	Yes
	Delay_R < 0.5 * (sdmmc_clk/4)	Yes	No
SDR12	Delay_R > 0.5 * (sdmmc_clk/4)	No	Yes
	Delay_R < 0.5 * (sdmmc_clk/4)	Yes	No

Related Information

Arria 10 Device Datasheet

Recommended Usage Guidelines for Card Read Threshold

1. The `cardthrctl` register must be set before setting the `cmd` register for a data read command.†
2. The `cardthrctl` register must not be set while a data transfer command is in progress.†
3. The `cardrdthreshold` field of the `cardthrctl` register must be set to at the least the block size of a single or multiblock transfer. A `cardrdthreshold` field setting greater than or equal to the block size of the read transfer ensures that the card clock does not stop in the middle of a block of data.†
4. If the round trip delay is greater than half of the card clock period, card read threshold must be enabled and the card threshold must be set as per guideline 3 to guarantee that the card clock does not stop in the middle of a block of data.†
5. If the `cardrdthreshold` field is set to less than the block size of the transfer, the host must ensure that the receive FIFO buffer never overflows during the read transfer. Overflow can cause the card clock from the controller to stop. The controller is not able to guarantee that the card clock does not stop during a read transfer.†

Note: If the `cardrdthreshold` field of the `cardthrctl` register, and the `rx_wmark` and `dw_dma_multiple_transaction_size` fields of the `fifoth` register are set incorrectly, the card clock might stop indefinitely, with no interrupts generated by the controller.†

Card Read Threshold Programming Sequence

Most cards, such as SDHC or SDXC, support block sizes that are either specified in the card or are fixed to 512 bytes. For SDIO, MMC, and standard capacity SD cards that support partial block read (READ_BL_PARTIAL set to 1 in the CSD register of the card device), the block size is variable and can be chosen by the application.†

(43) Delay_R = Delay_O + tODLY + Delay_I †

Where: †

Delay_O = `sdmmc_clk` to `sdmmc_cclk_out` delay (including I/O pin delay) †

Delay_I = Input I/O pin delay + routing delay to the input register †

tODLY = `sdmmc_cclk_out` to card output delay (varies across card manufactures and speed modes) †

For the delay numbers needed for above calculation, refer to Arria 10 Datasheet. †

To use the card read threshold feature effectively and to guarantee that the card clock does not stop because of a FIFO Full condition in the middle of a block of data being read from the card, follow these steps:†

1. Choose a block size that is a multiple of four bytes.†
2. Enable card read threshold feature. The card read threshold can be enabled only if the block size for the given transfer is less than or equal to the total depth of the FIFO buffer:†

   (block size / 4) ≤ 1024†
3. Choose the card read threshold value: †
   - If (block size / 4) ≥ 512, choose `cardrdthreshold` such that:†
     - `cardrdthreshold` ≤ (block size / 4) in bytes†
   - If (block size / 4) < 512, choose `cardrdthreshold` such that:†
     - `cardrdthreshold` = (block size / 4) in bytes†
4. Set the `dw_dma_multiple_transaction_size` field in the `fifoth` register to the number of transfers that make up a DMA transaction. For example, size = 1 means 4 bytes are moved. The possible values for the size are 1, 4, 8, 16, 32, 64, 128, and 256 transfers. Select the size so that the value (block size / 4) is evenly divided by the size.†
5. Set the `rx_wmark` field in the `fifoth` register to the size – 1.†

For example, if your block size is 512 bytes, legal values of `dw_dma_multiple_transaction_size` and `rx_wmark` are listed in the following table.

Table 14-37: Legal Values of dw_dma_multiple_transaction_size and rx_wmark for Block Size = 512†

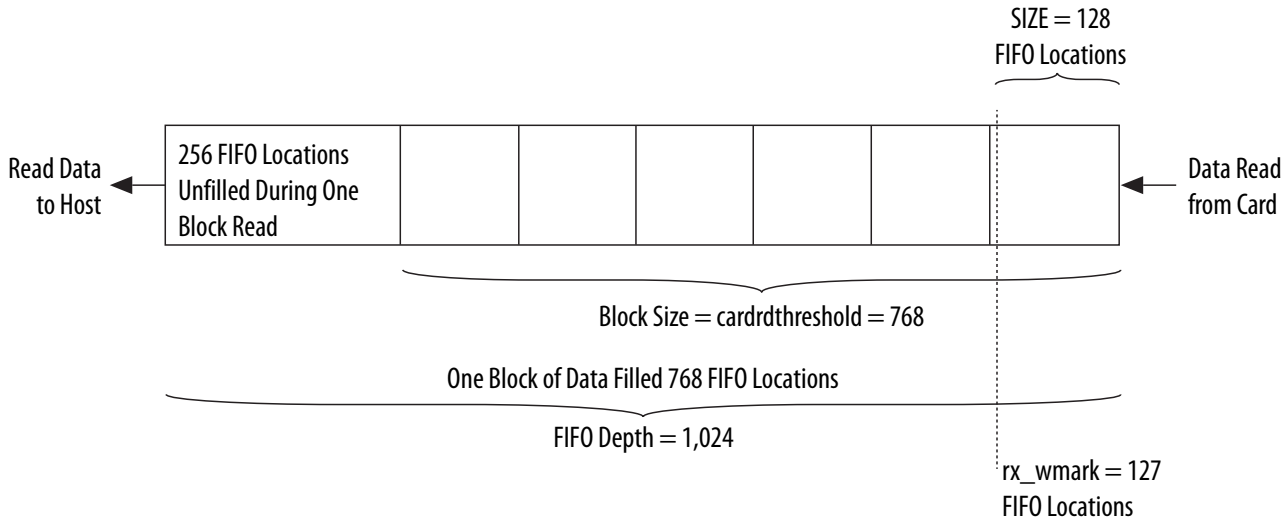
Block Size	dw_dma_multiple_transaction_size	rx_wmark
512	1	0
512	4	3
512	8	7
512	16	15
512	32	31
512	64	63
512	128	127

Card Read Threshold Programming Examples

This section shows examples of how to program the card read threshold.†

- Choose a block size that is a multiple of 4 (the number of bytes per FIFO location), and less than 4096 (1024 FIFO locations). For example, a block size of 3072 bytes is legal, because 3072 / 4 = 768 FIFO locations.†
- For DMA mode, choose the size so that block size is a multiple of the size. For example size = 128, where block size%size = 0.†
- Set the `rx_wmark` field = size – 1. For example, the `rx_wmark` field = 128 – 1 = 127.†
- Because block size > ½ FifoDepth, set the `cardrdthreshold` field to the block size. For example, the `cardrdthreshold` field = 3072 bytes.†

Figure 14-16: FIFO Buffer content when Card Read Threshold is set to 768†



Interrupt and Error Handling

This section describes how to use interrupts to handle errors. On power-on or reset, interrupts are disabled (the `int_enable` bit in the `ctrl` register is set to 0), and all the interrupts are masked (the `intmask` register default is 0). The controller error handling includes the following types of errors:

- Response and data timeout errors—For response time-outs, the host software can retry the command. For data time-outs, the controller has not received the data start bit from the card, so software can either retry the whole data transfer again or retry from a specified block onwards. By reading the contents of the `tcbcnt` register later, the software can decide how many bytes remain to be copied (read). †
- Response errors—Set to 1 when an error is received during response reception. If the response received is invalid, the software can retry the command. †
- Data errors—Set to 1 when a data receive error occurs. Examples of data receive errors: †
  - Data CRC†
  - Start bit not found †
  - End bit not found †

  These errors can be occur on any block. On receipt of an error, the software can issue an SD/SDIO STOP or SEND_IF_COND command, and retry the command for either the whole data or partial data.†
- Hardware locked error—Set to 1 when the controller cannot load a command issued by software. When software sets the `start_cmd` bit in the `cmd` register to 1, the controller tries to load the command. If the command buffer already contains a command, this error is raised, and the new command is discarded, requiring the software to reload the command.†
- FIFO buffer underrun/overrun error—If the FIFO buffer is full and software tries to write data to the FIFO buffer, an overrun error is set. Conversely, if the FIFO buffer is empty and the software tries to read data from the FIFO buffer, an underrun error is set. Before reading or writing data in the FIFO buffer, the software must read the FIFO buffer empty bit (`fifo_empty`) or FIFO buffer full bit (`fifo_full`) in the `status` register.†
- Data starvation by host timeout—This condition occurs when software does not service the FIFO buffer fast enough to keep up with the controller. Under this condition and when a read transfer is in process, the software must read data from the FIFO buffer, which creates space for further data reception. When a transmit operation is in process, the software must write data to fill the FIFO buffer so that the controller can write the data to the card.†
- CE-ATA errors†
- CRC error on command—If a CRC error is detected for a command, the CE-ATA card device does not send a response, and a response timeout is expected from the controller. The ATA layer is notified that an MMC transport layer error occurred.
- CRC error on command—If a CRC error is detected for a command, the CE-ATA card device does not send a response, and a response timeout is expected from the controller. The ATA layer is notified that an MMC transport layer error occurred.†
- Write operation—Any MMC transport layer error known to the card device causes an outstanding ATA command to be terminated. The ERR bits are set in the ATA status registers and the appropriate error code is sent to the Error Register (Error) on the ATA card device.†

  If the device interrupt bit of the CE-ATA card (the nIEN bit in the ATA control register) is set to 0, the CCS is sent to the host.†

  If the device interrupt bit is set to 1, the card device completes the entire data unit count if the host controller does not abort the ongoing transfer.†

  Note: During a multiple-block data transfer, if a negative CRC status is received from the card device, the data path signals a data CRC error to the BIU by setting the `dcrc` bit in the `rintsts` register to 1. It then continues further data transmission until all the bytes are transmitted.†
- Read operation—If MMC transport layer errors are detected by the host controller, the host completes the ATA command with an error status. The host controller can issue a CCSD command followed by a STOP_TRANSMISSION (CMD12) command to abort the read transfer. The host can also transfer the entire data unit count bytes without aborting the data transfer.†

Booting Operation for eMMC and MMC

This section describes how to set up the controller for eMMC and MMC boot operation.

Note: The BootROM and initial software do not use the boot partitions that are in the MMC card. This means that there is no boot partition support of the SD/MMC controller.

Boot Operation by Holding Down the CMD Line

The controller can boot from MMC4.3, MMC4.4, and MMC4.41 cards by holding down the CMD line.

For information about this boot method, refer to the following specifications, available on the JEDEC website:

- JEDEC Standard No. 84-A441
- JEDEC Standard No. 84-A44
- JEDEC Standard No. JESD84-A43

Related Information

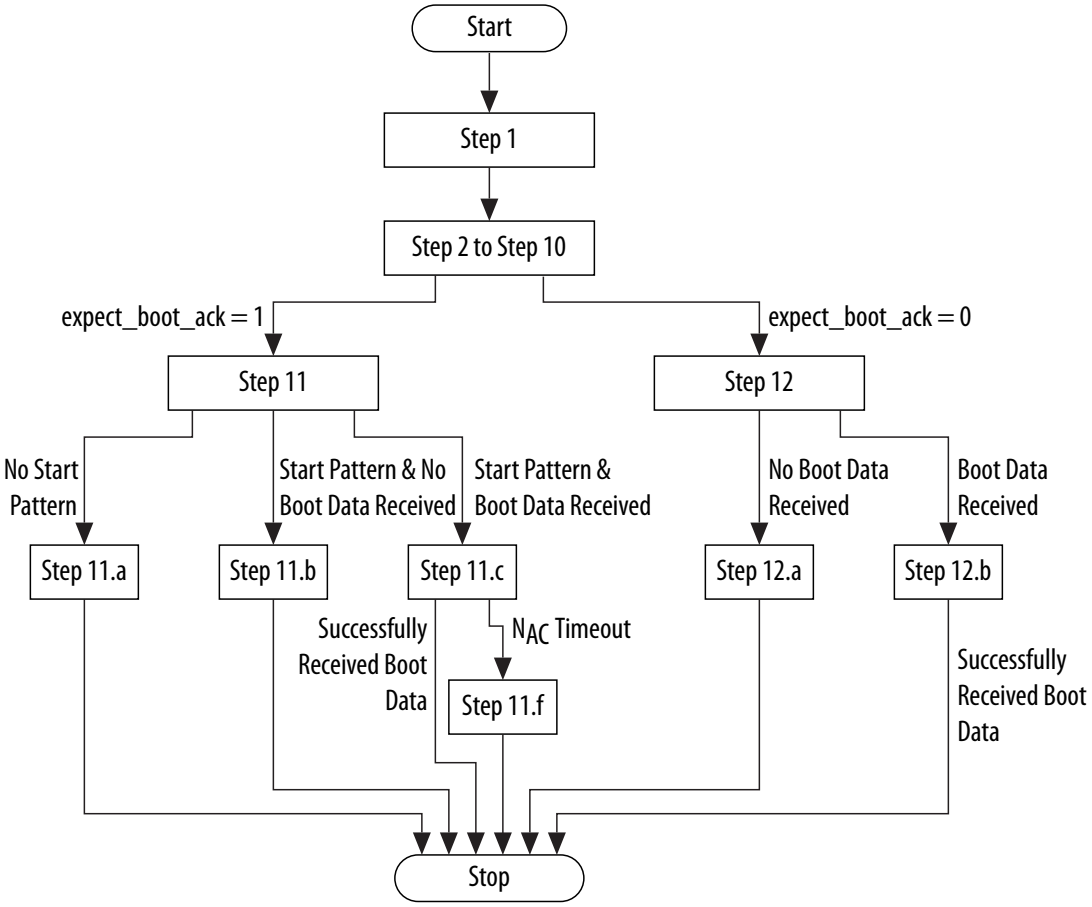
www.jedec.org

For more information about this boot method, refer to the following JEDEC Standards available on the JEDEC website: No. 84-A441, No. 84-A44, and No. JESD84-A43.

Boot Operation for eMMC Card Device

The following figure illustrates the steps to perform the boot process for eMMC card devices. The detailed steps are described following the flow chart.

Figure 14-17: Flow for eMMC Boot Operation†



1. The software driver performs the following checks: †
   - If the eMMC card device supports boot operation (the BOOT_PARTITION_ENABLE bit is set to 1 in the EXT_CSD register of the eMMC card).†
   - The BOOT_SIZE_MULT and BOOT_BUS_WIDTH values in the EXT_CSD register, to be used during the boot process.†
2. The software sets the following bits: †
   - Sets masks for interrupts, by setting the appropriate bits to 0 in the `intmask` register.†
   - Sets the global `int_enable` bit of the `ctrl` register to 1. Other bits in the `ctrl` register must be set to 0. †

   Note: Altera recommends that you write 0xFFFFFFFF to the `rintsts` and `idsts` registers to clear any pending interrupts before setting the `int_enable` bit. For internal DMA controller mode, the software driver needs to unmask all the relevant fields in the `idinten` register.†
3. If the software driver needs to use the internal DMA controller to transfer the boot data received, it must perform the following steps: †

- Set up the descriptors as described in *Internal DMA Controller Transmission Sequences* and *Internal DMA Controller Reception Sequences*". †
- Set the `use_internal_dmac` bit of the `ctrl` register to 1.†

4. Set the card device frequency to 400 kHz using the `clkdiv` registers. For more information, refer to Clock Setup.†
5. Set the `data_timeout` field of the `tmout` register equal to the card device total access time, N_{AC}. †
6. Set the `blksiz` register to 0x200 (512 bytes). †
7. Set the `bytcnt` register to a multiple of 128 KB, as indicated by the BOOT_SIZE_MULT value in the card device.†
8. Set the `rx_wmark` field in the `fifoth` register. Typically, the threshold value can be set to 512, which is half the FIFO buffer depth.†
9. Set the following fields in the `cmd` register:†
   - Initiate the command by setting `start_cmd` = 1†
   - Enable boot (`enable_boot`) = 1†
   - Expect boot acknowledge (`expect_boot_ack`): †
     - If a start-acknowledge pattern is expected from the card device, set `expect_boot_ack` to 1.†
     - If a start-acknowledge pattern is not expected from the card device, set `expect_boot_ack` to 0.†
   - Card number (`card_number`) = 0†
   - `data_expected` = 1†
   - Reset the remainder of `cmd` register bits to 0†
10. If no start-acknowledge pattern is expected from the card device (`expect_boot_ack` set to 0) proceed to **step 12**.†
11. This step handles the case where a start-acknowledge pattern is expected (`expect_boot_ack` was set to 1 in **step 9**).†
    a. If the Boot ACK Received interrupt is not received from the controller within 50 ms of initiating the command (**step 9**), the software driver must set the following `cmd` register fields: †
       - `start_cmd` = 1†
       - Disable boot (`disable_boot`)= 1†
       - `card_number` = 0 †
       - All other fields = 0†

       The controller generates a Command Done interrupt after deasserting the `CMD` pin of the card interface.†

       If internal DMA controller mode is used for the boot process, the controller performs the following steps after the Boot ACK Received timeout:†
       - The DMA descriptor is closed.†
       - The `ces` bit in the `idsts` register is set, indicating the Boot ACK Received timeout.†
       - The `ri` bit of the `idsts` register is not set.†

    b. If the Boot ACK Received interrupt is received, the software driver must clear this interrupt by writing 1 to the `ces` bit in the `idsts` register.†

       Within 0.95 seconds of the Boot ACK Received interrupt, the Boot Data Start interrupt must be received from the controller. If this does not occur, the software driver must write the following `cmd` register fields:†
       - `start_cmd` = 1†
       - disable_boot = 1†
       - card_number = 0†
       - All other fields = 0†

The controller generates a Command Done interrupt after deasserting the CMD pin of the card interface.†

If internal DMA controller mode is used for the boot process, the controller performs the following steps after the Boot ACK Received timeout:†

- The DMA descriptor is closed†
- The ces bit in the idsts register is set, indicating Boot Data Start timeout†
- The ri bit of the idsts register is not set†

c. If the Boot Data Start interrupt is received, it indicates that the boot data is being received from the card device. When the DMA engine is not in internal DMA controller mode, the software driver can then initiate a data read from the controller based on the rxdr interrupt bit in the rintsts register.†

In internal DMA controller mode, the DMA engine starts transferring the data from the FIFO buffer to the system memory as soon as the level set in the rx_wmark field of the fifoth register is reached.†

At the end of a successful boot data transfer from the card, the following interrupts are generated:†

- The cmd bit and dto bit in the rintsts register†
- The ri bit in the idsts register, in internal DMA controller mode only†

d. If an error occurs in the boot ACK pattern (0b010) or an EBE occurs: †

- The controller automatically aborts the boot process by pulling the CMD line high†
- The controller generates a Command Done interrupt†
- The controller does not generate a Boot ACK Received interrupt†
- The application aborts the boot transfer†

e. In internal DMA controller mode:†

- If the software driver creates more descriptors than required by the received boot data, the extra descriptors are not closed by the controller. Software cannot reuse the descriptors until they are closed.†
- If the software driver creates fewer descriptors than required by the received boot data, the controller generates a Descriptor Unavailable interrupt and does not transfer any further data to system memory.†

f. If N_{AC} is violated between data block transfers, the DRTO interrupt is asserted. In addition, if there is an error associated with the start or end bit, the SBE or EBE interrupt is also generated.†

The boot operation for eMMC card devices is complete. Do not execute the remaining (**step 12**).†

12. This step handles the case where no start-acknowledge pattern is expected (expect_boot_ack was set to 0 in **step 9**).†

a. If the Boot Data Start interrupt is not received from the controller within 1 second of initiating the command (**step 9**), the software driver must write the cmd register with the following fields:†

- start_cmd = 1†
- disable_boot = 1†
- card_number = 0†
- All other fields = 0†

The controller generates a Command Done interrupt after deasserting the CMD line of the card. In internal DMA controller mode, the descriptor is closed and the ces bit in the idsts register is set to 1, indicating a Boot Data Start timeout.†

b. If a Boot Data Start interrupt is received, it indicates that the boot data is being received from the card device. When the DMA engine is not in internal DMA controller mode, the software driver

can then initiate a data read from the controller based on the `rxdr` interrupt bit in the `rintsts` register.†

In internal DMA controller mode, the DMA engine starts transferring the data from the FIFO buffer to the system memory as soon as the level specified in the `rx_wmark` field of the `fifoth` register is reached.†

At the end of a successful boot data transfer from the card, the following interrupts are generated:†

- The `cmd` bit and `dto` bit in the `rintsts` register†
- The `ri` bit in the `idsts` register, in internal DMA controller mode only†

c. In internal DMA controller mode:†

- If the software driver creates more descriptors than required by the received boot data, the extra descriptors are not closed by the controller.†
- If the software driver creates fewer descriptors than required by the received boot data, the controller generates a Descriptor Unavailable interrupt and does not transfer any further data to system memory.†

The boot operation for eMMC card devices is complete.†

Related Information

- **Clock Setup** on page 14-46
  Refer to this section for information on how to set the card device frequency.
- **Internal DMA Controller Transmission Sequences** on page 14-58
  Refer to this section for information about the Internal DMA Controller Transmission Sequences.
- **Internal DMA Controller Reception Sequences** on page 14-59
  Refer to this section for information about the Internal DMA Controller Reception Sequences.

Boot Operation for Removable MMC4.3, MMC4.4 and MMC4.41 Cards

Removable MMC4.3, MMC4.4, and MMC4.41 Differences

Removable MMC4.3, MMC4.4, and MMC4.41 cards differ with respect to eMMC in that the controller is not aware whether these cards support the boot mode of operation when plugged in. Thus, the controller must: †

1. Discover these cards as it would discover MMC4.0/4.1/4.2 cards for the first time†
2. Know the card characteristics †
3. Decide whether to perform a boot operation or not†

Booting Removable MMC4.3, MMC4.4 and MMC4.41 Cards

For removable MMC4.3, MMC4.4 and MMC4.41 cards, the software driver must perform the following steps:†

1. Discover the card as described in *Enumerated Card Stack*.†
2. Read the EXT_CSD register of the card and examine the following fields: †
   - BOOT_PARTITION_ENABLE †
   - BOOT_SIZE_MULT†
   - BOOT_INFO †
3. If necessary, the software can manipulate the boot information in the card. †

Note: For more information, refer to "Access to Boot Partition" in the following specifications available on the JEDEC website:

- JEDEC Standard No. 84-A441
- JEDEC Standard No. 84-A44
- JEDEC Standard No. JESD84-A43

4. If the host processor needs to perform a boot operation at the next power-up cycle, it can manipulate the EXT_CSD register contents by using a SWITCH_FUNC command. †
5. After this step, the software driver must power down the card by writing to the `pwren` register. †
6. From here on, use the same steps as in *Alternative Boot Operation for eMMC Card Devices*.†

Related Information

- **Enumerated Card Stack** on page 14-43
  Refer to this section for more information on discovering removable MMC cards.
- **www.jedec.org**
  For more information, refer to "Access to Boot Partition" in the following specifications available on the JEDEC website: No. 84-A441, No. 84-A44, and No. JESD84-A43.
- **Alternative Boot Operation for eMMC Card Devices** on page 14-79
  Refer to this section for information about alternative boot operation steps.

Alternative Boot Operation

The alternative boot operation differs from the previous boot operation in that software uses the SD/SDIO GO_IDLE_STATE command to boot the card, rather than holding down the CMD line of the card. The alternative boot operation can be performed only if bit 0 in the BOOT_INFO register is set to 1. BOOT_INFO is located at offset 228 in the EXT_CSD registers. †

For detailed information about alternative boot operation, refer to the following specifications available on the JEDEC website:

- JEDEC Standard No. 84-A441
- JEDEC Standard No. 84-A44
- JEDEC Standard No. JESD84-A43

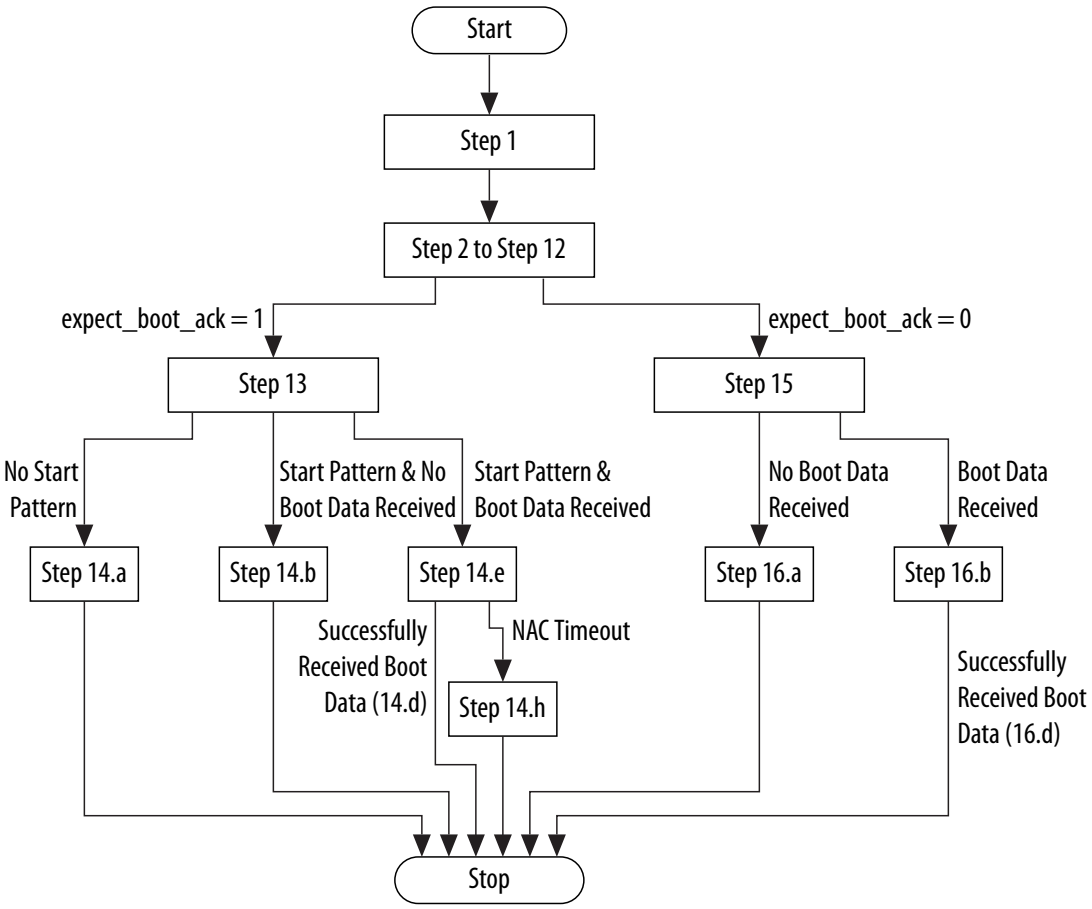
Related Information

www.jedec.org
For more information about alternative boot operation, refer to the following JEDEC Standards available on the JEDEC website: No. 84-A441, No. 84-A44, and No. JESD84-A43.

Alternative Boot Operation for eMMC Card Devices

The following figure illustrates the sequence of steps required to perform the alternative boot operation for eMMC card devices. The detailed steps are described following the flow chart.

Figure 14-18: Flow for eMMC Alternative Boot Operation†



1. The software driver checks:†
   - If the eMMC card device supports alternative boot operation (the BOOT_INFO bit is set to 1 in the eMMC card).†
   - The BOOT_SIZE_MULT and BOOT_BUS_WIDTH values in the card device to use during the boot process.†
2. The software sets the following bits: †
   - Sets masks for interrupts by resetting the appropriate bits to 0 in the `intmask` register.†
   - Sets the `int_enable` bit of the `ctrl` register to 1. Other bits in the `ctrl` register must be set to 0. †

   Note: Altera recommends writing 0xFFFFFFFF to the `rintsts` register and `idsts` register to clear any pending interrupts before setting the `int_enable` bit. For internal DMA controller mode, the software driver needs to unmask all the relevant fields in the `idinten` register.†
3. If the software driver needs to use the internal DMA controller to transfer the boot data received, it must perform the following actions: †
   - Set up the descriptors as described in *Internal DMA Controller Transmission Sequences* and *Internal DMA Controller Reception Sequences*. †
   - Set the use internal DMAC bit (`use_internal_dmac`) of the `ctrl` register to 1. †
4. Set the card device frequency to 400 kHz using the `clkdiv` registers. For more information, refer to *Clock Setup*. Ensure that the card clock is running.†

5. Wait for a time that ensures that at least 74 card clock cycles have occurred on the card interface.†
6. Set the `data_timeout` field of the `tmout` register equal to the card device total access time, N_{AC}. †
7. Set the `blksiz` register to 0x200 (512 bytes).†
8. Set the `bytcnt` register to multiples of 128K bytes, as indicated by the BOOT_SIZE_MULT value in the card device.†
9. Set the `rx_wmark` field in the `fifoth` register. Typically, the threshold value can be set to 512, which is half the FIFO buffer depth.†
10. Set the `cmdarg` register to 0xFFFFFFFA. †
11. Initiate the command, by setting the `cmd` register with the following fields: †
    - `start_cmd` = 1†
    - `enable_boot` = 1†
    - `expect_boot_ack`:†
      - If a start-acknowledge pattern is expected from the card device, set `expect_boot_ack` to 1.†
      - If a start-acknowledge pattern is not expected from the card device, set `expect_boot_ack` to 0. †
    - `card_number` = 0 †
    - `data_expected` = 1†
    - `cmd_index` = 0†
    - Set the remainder of `cmd` register bits to 0.†
12. If no start-acknowledge pattern is expected from the card device (`expect_boot_ack` set to 0) jump to **step 15**. †
13. Wait for the Command Done interrupt.†
14. This step handles the case where a start-acknowledge pattern is expected (`expect_boot_ack` was set to 1 in **step 11**).†
    a. If the Boot ACK Received interrupt is not received from the controller within 50 ms of initiating the command (**step 11**), the start pattern was not received. The software driver must discontinue the boot process and start with normal discovery. †

       If internal DMA controller mode is used for the boot process, the controller performs the following steps after the Boot ACK Received timeout:†
       - The DMA descriptor is closed.†
       - The `ces` bit in the `idsts` register is set to 1, indicating the Boot ACK Received timeout.†
       - The `ri` bit of the `idsts` register is not set.†

    b. If the Boot ACK Received interrupt is received, the software driver must clear this interrupt by writing 1 to it. †

       Within 0.95 seconds of the Boot ACK Received interrupt, the Boot Data Start interrupt must be received from the controller. If this does not occur, the software driver must discontinue the boot process and start with normal discovery.†

       If internal DMA controller mode is used for the boot process, the controller performs the following steps after the Boot ACK Received timeout: †
       - The DMA descriptor is closed.†
       - The `ces` bit in the `idsts` register is set to 1, indicating Boot Data Start timeout.†
       - The `ri` bit of the `idsts` register is not set.†

    c. If the Boot Data Start interrupt is received, it indicates that the boot data is being received from the card device. When the DMA engine is not in internal DMA controller mode, the software driver can then initiate a data read from the controller based on the `rxdr` interrupt bit in the `rintsts` register.†

In internal DMA controller mode, the DMA engine starts transferring the data from the FIFO buffer to the system memory as soon as the level specified in the `rx_wmark` field of the `fifoth` register is reached. †

d. The software driver must terminate the boot process by instructing the controller to send the SD/SDIO GO_IDLE_STATE command:†

- Reset the `cmdarg` register to 0.†
- Set the `start_cmd` bit of the `cmd` register to 1, and all other bits to 0.†

e. At the end of a successful boot data transfer from the card, the following interrupts are generated: †

- The `cmd` bit and `dto` bit in the `rintsts` register†
- The `ri` bit in the `idsts` register, in internal DMA controller mode only†

f. If an error occurs in the boot ACK pattern (0b010) or an EBE occurs: †

- The controller does not generate a Boot ACK Received interrupt. †
- The controller detects Boot Data Start and generates a Boot Data Start interrupt. †
- The controller continues to receive boot data. †
- The application must abort the boot process after receiving a Boot Data Start interrupt.†

g. In internal DMA controller mode: †

- If the software driver creates more descriptors than required by the received boot data, the extra descriptors are not closed by the controller. †
- If the software driver creates fewer descriptors than required by the received boot data, the controller generates a Descriptor Unavailable interrupt and does not transfer any further data to system memory.†

h. If N_{AC} is violated between data block transfers, a DRTO interrupt is asserted. Apart from this, if there is an error associated with the start or end bit, the SBE or EBE interrupt is also generated.†

The alternative boot operation for eMMC card devices is complete. Do not execute the remaining steps (**15** and **16**). †

15. Wait for the Command Done interrupt.†

16. This step handles the case where a start-acknowledge pattern is not expected (`expect_boot_ack` was set to 0 in **step 11**). †

a. If the Boot Data Start interrupt is not received from the controller within 1 second of initiating the command (**step 11**), the software driver must discontinue the boot process and start with normal discovery. † In internal DMA controller mode:†

- The DMA descriptor is closed.†
- The `ces` bit in the `idsts` register is set to 1, indicating Boot Data Start timeout.†
- The `ri` bit of the `idsts` register is not set.†

b. If a Boot Data Start interrupt is received, the boot data is being received from the card device. When the DMA engine is not in internal DMA controller mode, the software driver can then initiate a data read from the controller based on the `rxdr` interrupt bit in the `rintsts` register.†

In internal DMA controller mode, the DMA engine starts transferring the data from the FIFO buffer to the system memory as soon as the level specified in the `rx_wmark` field of the `fifoth` register is reached.†

c. The software driver must terminate the boot process by instructing the controller to send the SD/SDIO GO_IDLE_STATE (CMD0) command: †

- Reset the `cmdarg` register to 0.†
- Set the `start_cmd` bit in the `cmd` register to 1, and all other bits to 0.†

d. At the end of a successful boot data transfer from the card, the following interrupts are generated: †

- The cmd bit and dto bit in the rintsts register†
- The ri bit in the idsts register, in internal DMA controller mode only †

e. In internal DMA controller mode: †

- If the software driver creates more descriptors than required by the received boot data, the extra descriptors are not closed by the controller. †
- If the software driver creates fewer descriptors than required by the received boot data, the controller generates a Descriptor Unavailable interrupt and does not transfer any further data to system memory.†

The alternative boot operation for eMMC card devices is complete.†

Related Information

- **Clock Setup** on page 14-46
  Refer to this section for information on how to set the card device frequency.
- **Internal DMA Controller Transmission Sequences** on page 14-58
  Refer to this section for information about the Internal DMA Controller Transmission Sequences.
- **Internal DMA Controller Reception Sequences** on page 14-59
  Refer to this section for information about the Internal DMA Controller Reception Sequences.

Alternative Boot Operation for MMC4.3 Cards

Removable MMC4.3 Boot Mode Support

Removable MMC4.3 cards differ with respect to eMMC in that the controller is not aware whether these cards support the boot mode of operation. Thus, the controller must: †

1. Discover these cards as it would discover MMC4.0/4.1/4.2 cards for the first time †
2. Know the card characteristics †
3. Decide whether to perform a boot operation or not†

Discovering Removable MMC4.3 Boot Mode Support

For removable MMC4.3 cards, the software driver must perform the following steps: †

1. Discover the card as described in *Enumerated Card Stack*.†
2. Read the MMC card device's EXT_CSD registers and examine the following fields: †
   - BOOT_PARTITION_ENABLE †
   - BOOT_SIZE_MULT †
   - BOOT_INFO †

   Note: For more information, refer to "Access to Boot Partition" in JEDEC Standard No. JESD84-A43, available on the JEDEC website.†
3. If the host processor needs to perform a boot operation at the next power-up cycle, it can manipulate the contents of the EXT_CSD registers in the MMC card device, by using a SWITCH_FUNC command. †
4. After this step, the software driver must power down the card by writing to the pwren register. †
5. From here on, use the same steps as in *Alternative Boot Operation for eMMC Card Devices*. †

   Note: Ignore the EBE if it is generated during an abort scenario.

   If a boot acknowledge error occurs, the boot acknowledge received interrupt times out. †

In internal DMA controller mode, the application needs to depend on the descriptor close interrupt instead of the data done interrupt. †

Related Information

- **Enumerated Card Stack** on page 14-43
  Refer to this section for more information on discovering removable MMC cards.
- **www.jedec.org**
  For more information, refer to "Access to Boot Partition" in JEDEC Standard No. JESD84-A43, available on the JEDEC website.
- **Alternative Boot Operation for eMMC Card Devices** on page 14-79
  Refer to this section for information about alternative boot operation steps.

SD/MMC Controller Address Map and Register Definitions

For complete HPS address map and register definitions, refer to the **Arria 10 HPS Address Map and Register Definitions**.

Related Information

Error Checking and Correction Controller on page 11-1
For more information about how to program the ECC registers, refer to this chapter.

sdmmc Address Map

Module Instance	Base Address	End Address
i_sdmmc_sdmmc	0xFF808000	0xFF808FFF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
ctrl on page 14-95	0x0	32	RW	0x0	Control register
pwren on page 14-101	0x4	32	RW	0x0	Power Enable Register
clkdiv on page 14-102	0x8	32	RW	0x0	Clock Divider Register
clksrc on page 14-104	0xC	32	RW	0x0	Clock Source Register
clkena on page 14-105	0x10	32	RW	0x0	Clock Enable Register
tmout on page 14-107	0x14	32	RW	0xFFFFFF40	Timeout Register

Register	Offset	Width	Access	Reset Value	Description
ctype on page 14-108	0x18	32	RW	0x0	Card Type Register
blksiz on page 14-109	0x1C	32	RW	0x200	Block Size Register
bytcnt on page 14-110	0x20	32	RW	0x200	Byte Count Register
intmask on page 14-111	0x24	32	RW	0x0	Interrupt Mask Register
cmdarg on page 14-115	0x28	32	RW	0x0	Command Argument Register
cmd on page 14-116	0x2C	32	RW	0x20000000	Command Register
resp0 on page 14-127	0x30	32	RO	0x0	Response Register 0
resp1 on page 14-128	0x34	32	RO	0x0	Response Register 1
resp2 on page 14-129	0x38	32	RO	0x0	Response Register 2
resp3 on page 14-129	0x3C	32	RO	0x0	Response Register 3
mintsts on page 14-130	0x40	32	RO	0x0	Masked Interrupt Status Register
rintsts on page 14-134	0x44	32	RW	0x0	Raw Interrupt Status Register
status on page 14-139	0x48	32	RO	0x106	Status Register
fifoth on page 14-143	0x4C	32	RW	0x3FF0000	FIFO Threshold Watermark Register
cdetect on page 14-147	0x50	32	RO	0x1	Card Detect Register
wrtprt on page 14-148	0x54	32	RO	0x1	Card Detect Register
tcbcnt on page 14-149	0x5C	32	RO	0x0	Transferred CIU Card Byte Count Register
tbbcnt on page 14-150	0x60	32	RO	0x0	Transferred Host to BIU-FIFO Byte Count Register
debnce on page 14-151	0x64	32	RW	0xFFFFFF	Debounce Count Register
usrid on page 14-152	0x68	32	RW	0x7967797	User ID Register

Register	Offset	Width	Access	Reset Value	Description
verid on page 14-152	0x6C	32	RO	0x5342270A	Version ID Register
hcon on page 14-153	0x70	32	RO	0xC43081	Hardware Configuration Register
uhs_reg on page 14-155	0x74	32	RW	0x0	UHS-1 Register
rst_n on page 14-157	0x78	32	RW	0x1	Hardware Reset Register
bmod on page 14-158	0x80	32	RW	0x0	Bus Mode Register
pldmnd on page 14-161	0x84	32	WO	0x0	Poll Demand Register
dbaddr on page 14-162	0x88	32	RW	0x0	Descriptor List Base Address Register
idsts on page 14-163	0x8C	32	RW	0x0	Internal DMAC Status Register
idinten on page 14-168	0x90	32	RW	0x0	Internal DMAC Interrupt Enable Register
dscaddr on page 14-171	0x94	32	RO	0x0	Current Host Descriptor Address Register
bufaddr on page 14-172	0x98	32	RO	0x0	Current Buffer Descriptor Address Register
cardthrctl on page 14-172	0x100	32	RW	0x0	Card Threshold Control Register
back_end_power_r on page 14-174	0x104	32	RW	0x0	Back End Power Register
data on page 14-175	0x200	32	RW	0x0	Data FIFO Access
gpio on page 14-176	0x58	32	RW	0x0	General Purpose Input/ Output Register
uhs_reg_ext on page 14-177	0x108	32	RW	0x0	UHS Register Extention
emmc_ddr_reg on page 14-178	0x10C	32	RW	0x0	EMMC DDR Register

Register	Offset	Width	Access	Reset Value	Description
enable_shift on page 14-179	0x110	32	RW	0x0	Register to control the amount of shift on enables

sdmmc Summary

Base Address: 0xFF808000

Register Address Offset	Bit Fields															
i_sdmmc_sdmmc																
ctrl 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved						use_internal_dmac RW 0x0	enable_od_pullup RW 0x0	card_voltage_b RW 0x0				card_voltage_a RW 0x0			
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				ceata_device_interrupt_status RW 0x0	send_auto_stop_ccsd RW 0x0	send_ccsd RW 0x0	abort_read_data RW 0x0	send_irq_response RW 0x0	read_wait RW 0x0	dma_enable RW 0x0	int_enable RW 0x0	Reserved	dma_reset RW 0x0	fifo_reset RW 0x0	controller_reset RW 0x0
pwren 0x4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															power_enable RW 0x0
clkdiv 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	clk_divider3 RO 0x0								clk_divider2 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	clk_divider1 RO 0x0								clk_divider0 RW 0x0							

Register Address Offset	Bit Fields															
clksrc 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	clk_source RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	clk_source RO 0x0															
clkena 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															cclk_low_power RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															cclk_enable RW 0x0
tmout 0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	data_timeout RW 0xFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	data_timeout RW 0xFFFFFF								response_timeout RW 0x40							
ctype 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															card_width1 RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															card_width2 RW 0x0
blksiz 0x1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	block_size RW 0x200															

Register Address Offset	Bit Fields															
bytcnt 0x20	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	byte_count RW 0x200															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	byte_count RW 0x200															
intmask 0x24	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	sdio_int_mask RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ebe RW 0x0	acd RW 0x0	sbe RW 0x0	hle RW 0x0	frun RW 0x0	hto RW 0x0	drt RW 0x0	rto RW 0x0	dcrc RW 0x0	rcrc RW 0x0	rxdr RW 0x0	txdr RW 0x0	dto RW 0x0	cmd RW 0x0	re RW 0x0	cd RW 0x0
cmdarg 0x28	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cmd_arg RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cmd_arg RW 0x0															
cmd 0x2C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	start_cmd RW 0x0	Reserved	use_hold_reg RW 0x1	volt_switch RW 0x0	boot_mode RW 0x0	disable_boot RW 0x0	expect_boot_ack RW 0x0	enable_boot RW 0x0	ccs_expected RW 0x0	read_ceata_device RW 0x0	update_clock_registers_only RW 0x0	card_number RW 0x0				
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	send_initialization RW 0x0	stop_abort_cmd RW 0x0	wait_prvdata_complete RW 0x0	send_auto_stop RW 0x0	transfer_mode RW 0x0	read_write RW 0x0	data_expected RW 0x0	check_response_crc RW 0x0	response_length RW 0x0	response_expect RW 0x0	cmd_index RW 0x0					

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
resp0	response0 RO 0x0															
0x30	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	response0 RO 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
resp1	response1 RO 0x0															
0x34	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	response1 RO 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
resp2	response2 RO 0x0															
0x38	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	response2 RO 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
resp3	response3 RO 0x0															
0x3C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	response3 RO 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
mintsts	sdio_interrupt RO 0x0															
0x40	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ebe RO 0x0	acd RO 0x0	strerr RO 0x0	hlwerr RO 0x0	fifoovunerr RO 0x0	dshto RO 0x0	datardto RO 0x0	respto RO 0x0	datacrcerr RO 0x0	respcrcerr RO 0x0	rxfifodr RO 0x0	dttxfifodr RO 0x0	dt RO 0x0	cmd_done RO 0x0	resp RO 0x0	cd RO 0x0

Register Address Offset	Bit Fields															
rintsts 0x44	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	sdio_interrupt RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ebe RW 0x0	acd RW 0x0	sbe RW 0x0	hle RW 0x0	frun RW 0x0	hto RW 0x0	bds RW 0x0	bar RW 0x0	dcrc RW 0x0	rcrc RW 0x0	rxdr RW 0x0	txdr RW 0x0	dto RW 0x0	cmd RW 0x0	re RW 0x0	cd RW 0x0
status 0x48	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	dma_req RO 0x0	dma_ack RO 0x0	fifo_count RO 0x0													response_index RO 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	response_index RO 0x0					data_state_mc_busy RO 0x0	data_busy RO 0x0	data_3_status RO 0x1	command_fsm_states RO 0x0				fifo_full RO 0x0	fifo_empty RO 0x1	fifo_tx_watermark RO 0x1	fifo_rx_watermark RO 0x0
fifoth 0x4C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	dw_dma_multiple_transaction_size RW 0x0			rx_wmark RW 0x3FF											
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				tx_wmark RW 0x0											
cdetect 0x50	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															card_detect_n RO 0x1
wrtprt 0x54	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															write_protect RO 0x1

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
tcbcnt 0x5C	trans_card_byte_count RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	trans_card_byte_count RO 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
tbbcnt 0x60	trans_fifo_byte_count RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	trans_fifo_byte_count RO 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
debnce 0x64	Reserved								debounce_count RW 0xFFFFFF							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	debounce_count RW 0xFFFFFF															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
usrid 0x68	usr_id RW 0x7967797															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	usr_id RW 0x7967797															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
verid 0x6C	ver_id RO 0x5342270A															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ver_id RO 0x5342270A															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcon 0x70	Reserved				ac RO 0x0	aro RO 0x0	ncd RO 0x0		scfp RO 0x1	ihr RO 0x1	rios RO 0x0	dmadatawidth RO 0x1			dmaintf RO 0x0	
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	haddrwidth RO 0xC						hdatawidth RO 0x1			hbus RO 0x0	nc RO 0x0					ct RO 0x1

Register Address Offset	Bit Fields															
uhs_reg 0x74	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ddr_reg RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	volt_reg RW 0x0															
rst_n 0x78	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															card_reset RW 0x1
bmod 0x80	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved					pbl RO 0x0			de RW 0x0	dsl RW 0x0					fb RW 0x0	swr RW 0x0
pldmnd 0x84	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	pd WO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	pd WO 0x0															
dbaddr 0x88	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	sdl RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	sdl RW 0x0															
idsts 0x8C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															fsm RO 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	fsm RO 0x0			eb RO 0x0			ais RW 0x0	nis RW 0x0	Reserved		ces RW 0x0	du RW 0x0	Reserved	fbe RW 0x0	ri RW 0x0	ti RW 0x0

Register Address Offset	Bit Fields															
idinten 0x90	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						ai RW 0x0	ni RW 0x0	Reserved		ces RW 0x0	du RW 0x0	Reserved	fbe RW 0x0	ri RW 0x0	ti RW 0x0
dscaddr 0x94	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	hda RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hda RO 0x0															
bufaddr 0x98	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	hba RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hba RO 0x0															
cardthrctl 0x100	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved			cardrdthreshold RW 0x0												
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														busy_clr_int_en RW 0x0	cardrdthren RW 0x0
back_end_power_r 0x104	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	back_end_power RW 0x0															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
data 0x200	value RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	value RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gpio 0x58	Reserved								gpo RW 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	gpo RW 0x0								gpi RO 0x0							
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
uhs_reg_ext 0x108	ext_clk_mux_ctrl RW 0x0		clk_drv_phase_ctrl RW 0x0							clk_smpl_phase_ctrl RW 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	mmc_volt_reg RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
emmc_ddr_reg 0x10C	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															half_start_bit RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
enable_shift 0x110	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														enable_shift_card RW 0x0	

ctrl

Control register

Module Instance	Base Address	Register Address
i_sdmmc_sdmmc	0xFF808000	0xFF808000

Offset: 0x0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved						use_internal_dmac RW 0x0	enable_od_pullup RW 0x0	card_voltage_b RW 0x0				card_voltage_a RW 0x0			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				ceata_device_interrupt_status RW 0x0	send_auto_stop_ccsd RW 0x0	send_ccsd RW 0x0	abort_read_data RW 0x0	send_irq_response RW 0x0	read_wait RW 0x0	dma_enable RW 0x0	int_enable RW 0x0	Reserved	dma_reset RW 0x0	fifo_reset RW 0x0	controller_reset RW 0x0

ctrl Fields

Bit	Name	Description	Access	Reset
25	use_internal_dmac	Present only for the Internal DMAC configuration; else, it is reserved. 0-The host performs data transfers through the slave interface 1-Internal DMAC used for data transfer **Value** **Description** 0 DISABLED 1 ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
24	enable_od_pullup	External open-drain pullup 0-Disable 1-Enable Inverted value of this bit is output to ccmd_od_pullup_en_n port. When bit is set, command output always driven in open-drive mode; that is, DWC_mobile_storage drives either 0 or high impedance, and does not drive hard 1.	RW	0x0
23:20	card_voltage_b	Card regulator-B voltage setting; output to card_volt_b port. Optional feature; ports can be used as general-purpose outputs	RW	0x0
19:16	card_voltage_a	Card regulator-A voltage setting; output to card_volt_a port. Optional feature; ports can be used as general-purpose outputs	RW	0x0
11	ceata_device_interrupt_status	0-Interrupts not enabled in CE-ATA device 1-Interrupts are enabled in CE-ATA device **Value** **Description** 0 DISABLED 1 ENABLED	RW	0x0
10	send_auto_stop_ccsd	0-Clear bit if DWC_mobile_storage does not reset the bit 1-Send internally generated STOP after sending CCSD to CE-ATA device **Value** **Description** 0 DEASSERT 1 ASSERT	RW	0x0

Bit	Name	Description	Access	Reset
9	send_ccsd	0-Clear this bit if DWC_mobile_storage does not reset the bit 1-Send Command Completion Signal Disable (CCSD) to CE-ATA device **Value / Description** 0 DEASSERT 1 ASSERT	RW	0x0
8	abort_read_data	0-No change 1-After suspend command is issued during read-transfer, software polls card to find when suspend happened. Once suspend occurs,software sets bit to reset data state-machine, which is waiting for next block of data. Bit automatically clears once data statemachine resets to idle. Used in SDIO card suspend sequence. **Value / Description** 0 NOCHANGE 1 ACTIVATE	RW	0x0

Bit	Name	Description	Access	Reset
7	send_irq_response	0-No Change in this 1-Send auto IRQ response Bit automatically clears once response is sent. To wait for MMC card interrupts, host issues CMD40, and DWC_mobile_storage waits for interrupt response from MMC card(s). In meantime, if host wants DWC_mobile_storage to exit waiting for interrupt state, it can set this bit, at which time DWC_mobile_storage command state-machine sends CMD40 response on bus and returns to idle state. **Value** **Description** 0 NOCHANGE 1 ACTIVATE	RW	0x0
6	read_wait	0-Clear read wait 1-Assert read wait For sending read-wait to SDIO cards. **Value** **Description** 0 DEASSERT 1 ASSERT	RW	0x0
5	dma_enable	0-Disable DMA transfer mode 1-Enable DMA transfer mode Valid only if DWC_mobile_storage configured for External DMA interface.	RW	0x0

Bit	Name	Description	Access	Reset
4	int_enable	Global interrupt enable/disable bit: 0-Disable interrupts 1-Enable interrupts The int port is 1 only when this bit is 1 and one or more unmasked interrupts are set. **Value** **Description** 0 DISABLED 1 ENABLED	RW	0x0
2	dma_reset	0-No change 1-Reset internal DMA interface control logic To reset DMA interface, firmware should set bit to 1. This bit is auto-cleared after two AHB clocks. **Value** **Description** 0 NOCHANGE 1 ACTIVATE	RW	0x0
1	fifo_reset	0-No change 1-Reset to data FIFO To reset FIFO pointers To reset FIFO, firmware should set bit to 1. This bit is auto-cleared after completion of reset operation. **Value** **Description** 0 NOCHANGE 1 ACTIVATE	RW	0x0

Bit	Name	Description	Access	Reset
0	controller_reset	0-No change 1-Reset DWC_mobile_ storage controller To reset controller, firmware should set bit to 1. This bit is auto-cleared after two AHB and two cclk_in clock cycles. This resets: * BIU/CIU interface * CIU and state machines * abort_read_data, send_irq_response, and read_wait bits of Control register * start_cmd bit of Command register Does not affect any registers or DMA interface, or FIFO or host interrupts **Value** **Description** 0 NOCHANGE 1 ACTIVATE	RW	0x0

pwren

Power Enable Register

Module Instance	Base Address	Register Address
i_sdmmc_sdmmc	0xFF808000	0xFF808004

Offset: 0x4

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															power_enable RW 0x0

pwren Fields

Bit	Name	Description	Access	Reset
0	power_enable	Power on/off switch for up to 16 cards; for example, bit[0] controls card 0.Once power is turned on, firmware should wait for regulator/switch ramp-up time before trying to initialize card. 0-power off 1-power on Only NUM_CARDS number of bits are implemented.Bit values output to card_power_en port. Optional feature; ports can be used as general-purpose outputs. **Value** **Description** 0 OFF 1 ON	RW	0x0

clkdiv

Clock Divider Register

Module Instance	Base Address	Register Address
i_sdmmc_sdmmc	0xFF808000	0xFF808008

Offset: 0x8

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
clk_divider3 RO 0x0								clk_divider2 RO 0x0							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
clk_divider1 RO 0x0								clk_divider0 RW 0x0							

clkdiv Fields

Bit	Name	Description	Access	Reset
31:24	clk_divider3	Clock divider-3 value. Clock division is 2*n. For example, value of 0 means divide by 2*0 = 0 (no division, bypass), a value of 1 means divide by 2*1 = 2, a value of “ff” means divide by 2*255 = 510, and so on. In MMC-Ver3.3-only mode, bits not implemented because only one clock divider is supported.	RO	0x0
23:16	clk_divider2	Clock divider-2 value. Clock division is 2*n. For example, value of 0 means divide by 2*0 = 0 (no division, bypass), value of 1 means divide by 2*1 = 2, value of “ff” means divide by 2*255 = 510, and so on. In MMC-Ver3.3-only mode, bits not implemented because only one clock divider is supported.	RO	0x0
15:8	clk_divider1	Clock divider-1 value. Clock division is 2*n. For example, value of 0 means divide by 2*0 = 0 (no division, bypass), value of 1 means divide by 2*1 = 2, value of “ff” means divide by 2*255 = 510, and so on. In MMC-Ver3.3-only mode, bits not implemented because only one clock divider is supported	RO	0x0
7:0	clk_divider0	Clock divider-0 value. Clock division is 2*n. For example, value of 0 means divide by 2*0 = 0 (no division, bypass), value of 1 means divide by 2*1 = 2, value of “ff” means divide by 2*255 = 510, and so on.	RW	0x0

clksrc

Clock Source Register

Module Instance	Base Address	Register Address
i_sdmmc_sdmmc	0xFF808000	0xFF80800C

Offset: 0xC

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
clk_source RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
clk_source RO 0x0															

clksrc Fields

Bit	Name	Description	Access	Reset
31:0	clk_source	Clock divider source for up to 16 SD cards supported. Each card has two bits assigned to it. For example, bits[1:0] assigned for card-0, which maps and internally routes clock divider[3:0] outputs to cclk_out[15:0] pins, depending on bit value. 00 Clock divider 0 01 Clock divider 1 10 Clock divider 2 11 Clock divider 3 In MMC-Ver3.3-only controller, only one clock divider supported. The cclk_out is always from clock divider 0, and this register is not implemented. **Value** **Description** 0 CLKDIV0	RO	0x0

clkena

Clock Enable Register

Module Instance	Base Address	Register Address
i_sdmmc_sdmmc	0xFF808000	0xFF808010

Offset: 0x10

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															cclk_low_power RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															cclk_enable RW 0x0

clkena Fields

Bit	Name	Description	Access	Reset
16	cclk_low_power	Low-power control for up to 16 SD card clocks and one MMC card clock supported. 0-Non-low-power mode 1-Low-power mode; stop clock when card in IDLE (should be normally set to only MMC and SD memory cards; for SDIO cards, if interrupts must be detected, clock should not be stopped). In MMC-Ver3.3-only mode, since there is only one cclk_out, only cclk_low_power[0] is used. **Value** — **Description** 0 — DISABLED 1 — ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
0	cclk_enable	Clock-enable control for up to 16 SD card clocks and one MMC card clock supported. 0-Clock disabled 1-Clock enabled In MMC-Ver3.3-only mode, since there is only one cclk_out, only cclk_enable[0] is used. **Value** **Description** 0 DISABLED 1 ENABLED	RW	0x0

tmout

Timeout Register

Module Instance	Base Address	Register Address
i_sdmmc_sdmmc	0xFF808000	0xFF808014

Offset: 0x14

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
data_timeout RW 0xFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
data_timeout RW 0xFFFFFF								response_timeout RW 0x40							

tmout Fields

Bit	Name	Description	Access	Reset
31:8	data_timeout	Value for card Data Read Timeout; same value also used for Data Starvation by Host timeout. The timeout counter is started only after the card clock is stopped.Value is in number of card output clocks cclk_out of selected card. Note: The software timer should be used if the timeout value is in the order of 100 ms. In this case, read data timeout interrupt needs to be disabled.	RW	0xFFFFFF
7:0	response_timeout	Response timeout value. Value is in number of card output clocks cclk_out.	RW	0x40

ctype

Card Type Register

Module Instance	Base Address	Register Address
i_sdmmc_sdmmc	0xFF808000	0xFF808018

Offset: 0x18

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															card_width1 RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															card_width2 RW 0x0

ctype Fields

Bit	Name	Description	Access	Reset
16	card_width1	One bit per card indicates if card is 8-bit: 0-Non 8-bit mode 1-8-bit mode Bit[31] corresponds to card[15]; bit[16] corresponds to card[0]. **Value** / **Description** 0 NON8BIT 1 MODE8BIT	RW	0x0
0	card_width2	One bit per card indicates if card is 1-bit or 4-bit: 0-1-bit mode 1-4-bit mode Bit[15] corresponds to card[15], bit[0] corresponds to card[0]. Only NUM_CARDS*2 number of bits are implemented. **Value** / **Description** 0 MODE1BIT 1 MODE4BIT	RW	0x0

blksiz

Block Size Register

Module Instance	Base Address	Register Address
i_sdmmc_sdmmc	0xFF808000	0xFF80801C

Offset: 0x1C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
block_size RW 0x200															

blksiz Fields

Bit	Name	Description	Access	Reset
15:0	block_size	Block size	RW	0x200

bytcnt

Byte Count Register

Module Instance	Base Address	Register Address
i_sdmmc_sdmmc	0xFF808000	0xFF808020

Offset: 0x20

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
byte_count RW 0x200															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
byte_count RW 0x200															

bytcnt Fields

Bit	Name	Description	Access	Reset
31:0	byte_count	Number of bytes to be transferred; should be integer multiple of Block Size for block transfers. For undefined number of byte transfers, byte count should be set to 0. When byte count is set to 0, it is responsibility of host to explicitly send stop/abort command to terminate data transfer.	RW	0x200

intmask

Interrupt Mask Register

Module Instance	Base Address	Register Address
i_sdmmc_sdmmc	0xFF808000	0xFF808024

Offset: 0x24

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
sdio_int_mask RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ebe RW 0x0	acd RW 0x0	sbe RW 0x0	hle RW 0x0	frun RW 0x0	hto RW 0x0	drt RW 0x0	rto RW 0x0	dcrc RW 0x0	rcrc RW 0x0	rxdr RW 0x0	txdr RW 0x0	dto RW 0x0	cmd RW 0x0	re RW 0x0	cd RW 0x0

intmask Fields

Bit	Name	Description	Access	Reset
31:16	sdio_int_mask	Mask SDIO interrupts One bit for each card. Bit[31] corresponds to card[15], and bit[16] corresponds to card[0]. When masked, SDIO interrupt detection for that card is disabled. A 0 masks an interrupt, and 1 enables an interrupt. In MMC-Ver3.3-only mode, these bits are always 0. **Value: Description** 0: DISABLED 1: ENABLED	RW	0x0
15	ebe	Bits used to mask unwanted interrupts. Value of 0 masks interrupts, value of 1 enables interrupt. **Value: Description** 0: MASK 1: NOMASK	RW	0x0
14	acd	Bits used to mask unwanted interrupts. Value of 0 masks interrupts, value of 1 enables interrupt. **Value: Description** 0: MASK 1: NOMASK	RW	0x0
13	sbe	Bits used to mask unwanted interrupts. Value of 0 masks interrupts, value of 1 enables interrupt. **Value: Description** 0: MASK 1: NOMASK	RW	0x0

Bit	Name	Description	Access	Reset
12	hle	Bits used to mask unwanted interrupts. Value of 0 masks interrupts, value of 1 enables interrupt. **Value** **Description** 0 MASK 1 NOMASK	RW	0x0
11	frun	Bits used to mask unwanted interrupts. Value of 0 masks interrupts, value of 1 enables interrupt. **Value** **Description** 0 MASK 1 NOMASK	RW	0x0
10	hto	Bits used to mask unwanted interrupts. Value of 0 masks interrupts, value of 1 enables interrupt. **Value** **Description** 0 MASK 1 NOMASK	RW	0x0
9	drt	Bits used to mask unwanted interrupts. Value of 0 masks interrupts, value of 1 enables interrupt. **Value** **Description** 0 MASK 1 NOMASK	RW	0x0
8	rto	Bits used to mask unwanted interrupts. Value of 0 masks interrupts, value of 1 enables interrupt. **Value** **Description** 0 MASK 1 NOMASK	RW	0x0

Bit	Name	Description	Access	Reset
7	dcrc	Bits used to mask unwanted interrupts. Value of 0 masks interrupts, value of 1 enables interrupt. **Value** **Description** 0 MASK 1 NOMASK	RW	0x0
6	rcrc	Bits used to mask unwanted interrupts. Value of 0 masks interrupts, value of 1 enables interrupt. **Value** **Description** 0 MASK 1 NOMASK	RW	0x0
5	rxdr	Bits used to mask unwanted interrupts. Value of 0 masks interrupts, value of 1 enables interrupt. **Value** **Description** 0 MASK 1 NOMASK	RW	0x0
4	txdr	Bits used to mask unwanted interrupts. Value of 0 masks interrupts, value of 1 enables interrupt. **Value** **Description** 0 MASK 1 NOMASK	RW	0x0
3	dto	Bits used to mask unwanted interrupts. Value of 0 masks interrupts, value of 1 enables interrupt. **Value** **Description** 0 MASK 1 NOMASK	RW	0x0

Bit	Name	Description	Access	Reset
2	cmd	Bits used to mask unwanted interrupts. Value of 0 masks interrupts, value of 1 enables interrupt. Value: Description 0: MASK 1: NOMASK	RW	0x0
1	re	Bits used to mask unwanted interrupts. Value of 0 masks interrupts, value of 1 enables interrupt. Value: Description 0: MASK 1: NOMASK	RW	0x0
0	cd	Bits used to mask unwanted interrupts. Value of 0 masks interrupts, value of 1 enables interrupt. Value: Description 0: MASK 1: NOMASK	RW	0x0

cmdarg

Command Argument Register

Module Instance	Base Address	Register Address
i_sdmmc_sdmmc	0xFF808000	0xFF808028

Offset: 0x28

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cmd_arg RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cmd_arg RW 0x0															

cmdarg Fields

Bit	Name	Description	Access	Reset
31:0	cmd_arg	Value indicates command argument to be passed to card	RW	0x0

cmd

Command Register

Module Instance	Base Address	Register Address
i_sdmmc_sdmmc	0xFF808000	0xFF80802C

Offset: 0x2C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
start_cmd RW 0x0	Reserved	use_hold_reg RW 0x1	volt_switch RW 0x0	boot_mode RW 0x0	disable_boot RW 0x0	expect_boot_ack RW 0x0	enable_boot RW 0x0	ccs_expected RW 0x0	read_ceata_device RW 0x0	update_clock_registers_only RW 0x0	card_number RW 0x0				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
send_initial-ization RW 0x0	stop_abort_cmd RW 0x0	wait_prvdata_complete RW 0x0	send_auto_stop RW 0x0	transfer_mode RW 0x0	read_write RW 0x0	data_expected RW 0x0	check_response_crc RW 0x0	response_length RW 0x0	response_expect RW 0x0	cmd_index RW 0x0					

cmd Fields

Bit	Name	Description	Access	Reset
31	start_cmd	Start command. Once command is taken by CIU, bit is cleared. When bit is set, host should not attempt to write to any command registers. If write is attempted, hardware lock error is set in raw interrupt register. Once command is sent and response is received from SD_MMC_CEATA cards, Command Done bit is set in raw interrupt register. **Value** — **Description** 0 — NOSTART 1 — START	RW	0x0

Bit	Name	Description	Access	Reset
29	use_hold_reg	Use Hold Register 0 - CMD and DATA sent to card bypassing HOLD Register 1 - CMD and DATA sent to card through the HOLD Register **Value: Description** 0: BYPASS 1: NOBYPASS	RW	0x1
28	volt_switch	Voltage switch bit 0 - No voltage switching 1 - Voltage switching enabled; must be set for CMD11 only **Value: Description** 0: NOVOLTSW 1: VOLTSW	RW	0x0
27	boot_mode	Boot Mode 0 - Mandatory Boot operation 1 - Alternate Boot operation **Value: Description** 0: MANDATORY 1: ALTERNATE	RW	0x0
26	disable_boot	Disable Boot. When software sets this bit along with start_cmd, CIU terminates the boot operation. Do NOT set disable_boot and enable_boot together. **Value: Description** 0: NOTERMBOOT 1: TERMBOOT	RW	0x0

Bit	Name	Description	Access	Reset
25	expect_boot_ack	Expect Boot Acknowledge. When Software sets this bit along with enable_boot, CIU expects a boot acknowledge start pattern of 0-1-0 from the selected card. **Value** **Description** 0 NOBOOTACK 1 BOOTACK	RW	0x0
24	enable_boot	Enable Bootthis bit should be set only for mandatory boot mode. When Software sets this bit along with start_cmd, CIU starts the boot sequence for the corresponding card by asserting the CMD line low. Do NOT set disable_boot and enable_boot together. **Value** **Description** 0 DISABLED 1 ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
23	ccs_expected	0-Interrupts are not enabled in CE-ATA device (nIEN = 1 in ATA control register), or command does not expect CCS from device 1-Interrupts are enabled in CE-ATA device (nIEN = 0), and RW_BLK command expects command completion signal from CE-ATA device If the command expects Command Completion Signal (CCS) from the CE-ATA device, the software should set this control bit.DWC_mobile_storage sets Data Transfer Over (DTO) bit in RINTSTS register and generates interrupt to host if Data Transfer Over interrupt is not masked. **Value** **Description** 0 DISABLED 1 ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
22	read_ceata_device	0-Host is not performing read access (RW_REG or RW_BLK) towards CE-ATA device 1-Host is performing read access (RW_REG or RW_BLK) towards CE-ATA device Software should set this bit to indicate that CE-ATA device is being accessed for read transfer. This bit is used to disable read data timeout indication while performing CE-ATA read transfers.Maximum value of I/O transmission delay can be no less than 10 seconds. DWC_mobile_storage should not indicate read data timeout while waiting for data from CE-ATA device. **Value** **Description** 0 NORD 1 RD	RW	0x0

Bit	Name	Description	Access	Reset
21	update_clock_ registers_only	0-Normal command sequence 1-Do not send commands, just update clock register value into card clock domain Following register values transferred into card clock domain: CLKDIV, CLRSRC, CLKENA. Changes card clocks (change frequency, truncate off or on, and set low-frequency mode); provided in order to change clock frequency or stop clock without having to send command to cards.During normal command sequence, when update_clock_ registers_only = 0, following control registers are transferred from BIU to CIU: CMD, CMDARG, TMOUT, CTYPE,BLKSIZ, BYTCNT. CIU uses new register values for new command sequence to card(s).When bit is set, there are no Command Done interrupts because no command is sent to SD_MMC_CEATA cards. **Value** **Description** 0 NORMCMD 1 UPDATCLKREG	RW	0x0
20:16	card_number	Card number in use. Represents physical slot number of card being accessed. In MMC-Ver3.3-only mode, up to 30 cards are supported; in SD-only mode, up to 16 cards are supported. Registered version of this is reflected on dw_dma_card_ num and ge_dma_card_num ports, which can be used to create separate DMA requests, if needed. In addition, in SD mode this is used to mux or demux signals from selected card because each card is interfaced to DWC_mobile_storage by separate bus.	RW	0x0

Bit	Name	Description	Access	Reset
15	send_initialization	0-Do not send initialization sequence (80 clocks of 1) before sending this command 1-Send initialization sequence before sending this command After power on, 80 clocks must be sent to card for initialization before sending any commands to card. Bit should be set while sending first command to card so that controller will initialize clocks before sending command to card. This bit should not be set for either of the boot modes (alternate or mandatory). **Value** **Description** 0 NOINIT 1 INIT	RW	0x0

Bit	Name	Description	Access	Reset
14	stop_abort_cmd	0-Neither stop nor abort command to stop current data transfer in progress. If abort is sent to function-number currently selected or not in data-transfer mode, then bit should be set to 0. 1-Stop or abort command intended to stop current data transfer in progress. When open-ended or predefined data transfer is in progress, and host issues stop or abort command to stop data transfer, bit should be set so that command/data state-machines of CIU can return correctly to idle state. This is also applicable for Boot mode transfers. To Abort boot mode, this bit should be set along with CMD[26] = disable_boot. **Value** **Description** 0 NOSTOPABRT 1 STOPABRT	RW	0x0

Bit	Name	Description	Access	Reset
13	wait_prvdata_complete	0-Send command at once, even if previous data transfer has not completed 1-Wait for previous data transfer completion before sending command The wait_prvdata_complete = 0 option typically used to query status of card during data transfer or to stop current data transfer;card_number should be same as in previous command. **Value: Description** 0: NOWAIT 1: WAIT	RW	0x0
12	send_auto_stop	0-No stop command sent at end of data transfer 1-Send stop command at end of data transfer When set, DWC_mobile_storage sends stop command to SD_MMC_CEATA cards at end of data transfer. * when send_auto_stop bit should be set, since some data transfers do not need explicit stop commands * open-ended transfers that software should explicitly send to stop command Additionally, when “resume” is sent to resume suspended memory access of SD-Combo card bit should be set correctly if suspended data transfer needs send_auto_stop.Don't care if no data expected from card. **Value: Description** 0: NOSEND 1: SEND	RW	0x0

Bit	Name	Description	Access	Reset
11	transfer_mode	0-Block data transfer command 1-Stream data transfer command Don't care if no data expected. **Value** **Description** 0 BLK 1 STR	RW	0x0
10	read_write	0-Read from card 1-Write to card Don't care if no data expected from card. **Value** **Description** 0 RD 1 WR	RW	0x0
9	data_expected	0-No data transfer expected (read/write) 1-Data transfer expected (read/write) **Value** **Description** 0 NODATXFEREXP 1 DATAXFEREXP	RW	0x0
8	check_response_crc	0-Do not check response CRC 1-Check response CRC Some of command responses do not return valid CRC bits. Software should disable CRC checks for those commands in order to disable CRC checking by controller. **Value** **Description** 0 NOCHK 1 CHK	RW	0x0

Bit	Name	Description	Access	Reset
7	response_length	0-Short response expected from card 1-Long response expected from card **Value** **Description** 0 SHORT 1 LONG	RW	0x0
6	response_expect	0-No response expected from card 1-Response expected from card **Value** **Description** 0 RESP 1 NORESP	RW	0x0
5:0	cmd_index	Command index	RW	0x0

resp0

Response Register 0

Module Instance	Base Address	Register Address
i_sdmmc_sdmmc	0xFF808000	0xFF808030

Offset: 0x30

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
response0 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
response0 RO 0x0															

resp0 Fields

Bit	Name	Description	Access	Reset
31:0	response0	Bit[31:0] of response	RO	0x0

resp1

Response Register 1

Module Instance	Base Address	Register Address
i_sdmmc_sdmmc	0xFF808000	0xFF808034

Offset: 0x34

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
response1 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
response1 RO 0x0															

resp1 Fields

Bit	Name	Description	Access	Reset
31:0	response1	Register represents bit[63:32] of long response. When CIU sends auto-stop command, then response is saved in register. Response for previous command sent by host is still preserved in Response 0 register. Additional auto-stop issued only for data transfer commands, and response type is always “short” for them.	RO	0x0

resp2

Response Register 2

Module Instance	Base Address	Register Address
i_sdmmc_sdmmc	0xFF808000	0xFF808038

Offset: 0x38

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
response2 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
response2 RO 0x0															

resp2 Fields

Bit	Name	Description	Access	Reset
31:0	response2	Bit[95:64] of long response	RO	0x0

resp3

Response Register 3

Module Instance	Base Address	Register Address
i_sdmmc_sdmmc	0xFF808000	0xFF80803C

Offset: 0x3C

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
response3 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
response3 RO 0x0															

resp3 Fields

Bit	Name	Description	Access	Reset
31:0	response3	Bit[127:96] of long response	RO	0x0

mintsts

Masked Interrupt Status Register

Module Instance	Base Address	Register Address
i_sdmmc_sdmmc	0xFF808000	0xFF808040

Offset: 0x40

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
sdio_interrupt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ebe RO 0x0	acd RO 0x0	strerr RO 0x0	hlwerr RO 0x0	fifovunerr RO 0x0	dshto RO 0x0	datardto RO 0x0	resptо RO 0x0	datacrcerr RO 0x0	respcrcerr RO 0x0	rxfifodr RO 0x0	dttxfifodr RO 0x0	dt RO 0x0	cmd_done RO 0x0	resp RO 0x0	cd RO 0x0

mintsts Fields

Bit	Name	Description	Access	Reset
31:16	sdio_interrupt	Interrupt from SDIO card; one bit for each card. Bit[31] corresponds to Card[15], and bit[16] is for Card[0]. SDIO interrupt for card enabled only if corresponding sdio_int_mask bit is set in Interrupt mask register (mask bit 1 enables interrupt; 0 masks interrupt). 0-No SDIO interrupt from card 1-SDIO interrupt from card In MMC-Ver3.3-only mode, bits always 0. **Value** **Description** 0 INACTIVE 1 ACTIVE	RO	0x0
15	ebe	Interrupt enabled only if corresponding bit in interrupt mask register is set. **Value** **Description** 0 MASK 1 NOMASK	RO	0x0
14	acd	Interrupt enabled only if corresponding bit in interrupt mask register is set. **Value** **Description** 0 MASK 1 NOMASK	RO	0x0

Bit	Name	Description	Access	Reset
13	strerr	Interrupt enabled only if corresponding bit in interrupt mask register is set. Value: 0 = MASK; 1 = NOMASK	RO	0x0
12	hlwerr	Interrupt enabled only if corresponding bit in interrupt mask register is set. Value: 0 = MASK; 1 = NOMASK	RO	0x0
11	fifoovunerr	Interrupt enabled only if corresponding bit in interrupt mask register is set. Value: 0 = MASK; 1 = NOMASK	RO	0x0
10	dshto	Interrupt enabled only if corresponding bit in interrupt mask register is set. Value: 0 = MASK; 1 = NOMASK	RO	0x0
9	datardto	Interrupt enabled only if corresponding bit in interrupt mask register is set. Value: 0 = MASK; 1 = NOMASK	RO	0x0

Bit	Name	Description	Access	Reset
8	respto	Interrupt enabled only if corresponding bit in interrupt mask register is set. Value: 0 = MASK; 1 = NOMASK	RO	0x0
7	datacrcerr	Interrupt enabled only if corresponding bit in interrupt mask register is set. Value: 0 = MASK; 1 = NOMASK	RO	0x0
6	respcrcerr	Interrupt enabled only if corresponding bit in interrupt mask register is set. Value: 0 = MASK; 1 = NOMASK	RO	0x0
5	rxfifodr	Interrupt enabled only if corresponding bit in interrupt mask register is set. Value: 0 = MASK; 1 = NOMASK	RO	0x0
4	dttxfifodr	Interrupt enabled only if corresponding bit in interrupt mask register is set. Value: 0 = MASK; 1 = NOMASK	RO	0x0

Bit	Name	Description	Access	Reset
3	dt	Interrupt enabled only if corresponding bit in interrupt mask register is set. Value: 0 = MASK; 1 = NOMASK	RO	0x0
2	cmd_done	Interrupt enabled only if corresponding bit in interrupt mask register is set. Value: 0 = MASK; 1 = NOMASK	RO	0x0
1	resp	Interrupt enabled only if corresponding bit in interrupt mask register is set. Value: 0 = MASK; 1 = NOMASK	RO	0x0
0	cd	Interrupt enabled only if corresponding bit in interrupt mask register is set. Value: 0 = MASK; 1 = NOMASK	RO	0x0

rintsts

Raw Interrupt Status Register

Module Instance	Base Address	Register Address
i_sdmmc_sdmmc	0xFF808000	0xFF808044

Offset: 0x44

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
sdio_interrupt RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ebe RW 0x0	acd RW 0x0	sbe RW 0x0	hle RW 0x0	frun RW 0x0	hto RW 0x0	bds RW 0x0	bar RW 0x0	dcrc RW 0x0	rcrc RW 0x0	rxdr RW 0x0	txdr RW 0x0	dto RW 0x0	cmd RW 0x0	re RW 0x0	cd RW 0x0

rintsts Fields

Bit	Name	Description	Access	Reset
31:16	sdio_interrupt	Interrupt from SDIO card; one bit for each card. Bit[31] corresponds to Card[15], and bit[16] is for Card[0]. Writes to these bits clear them. Value of 1 clears bit and 0 leaves bit intact. 0-No SDIO interrupt from card 1-SDIO interrupt from card In MMC-Ver3.3-only mode, bits always 0. Bits are logged regardless of interrupt-mask status. **Value** **Description** 0 INACTIVE 1 ACTIVE	RW	0x0
15	ebe	Writes to bits clear status bit. Value of 1 clears status bit, and value of 0 leaves bit intact. Bits are logged regardless of interrupt mask status. **Value** **Description** 0 INACTIVE 1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
14	acd	Writes to bits clear status bit. Value of 1 clears status bit, and value of 0 leaves bit intact. Bits are logged regardless of interrupt mask status. **Value** **Description** 0 INACTIVE 1 ACTIVE	RW	0x0
13	sbe	Writes to bits clear status bit. Value of 1 clears status bit, and value of 0 leaves bit intact. Bits are logged regardless of interrupt mask status. **Value** **Description** 0 INACTIVE 1 ACTIVE	RW	0x0
12	hle	Writes to bits clear status bit. Value of 1 clears status bit, and value of 0 leaves bit intact. Bits are logged regardless of interrupt mask status. **Value** **Description** 0 INACTIVE 1 ACTIVE	RW	0x0
11	frun	Writes to bits clear status bit. Value of 1 clears status bit, and value of 0 leaves bit intact. Bits are logged regardless of interrupt mask status. **Value** **Description** 0 INACTIVE 1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
10	hto	Writes to bits clear status bit. Value of 1 clears status bit, and value of 0 leaves bit intact. Bits are logged regardless of interrupt mask status. **Value** **Description** 0 INACTIVE 1 ACTIVE	RW	0x0
9	bds	Writes to bits clear status bit. Value of 1 clears status bit, and value of 0 leaves bit intact. Bits are logged regardless of interrupt mask status. **Value** **Description** 0 INACTIVE 1 ACTIVE	RW	0x0
8	bar	Writes to bits clear status bit. Value of 1 clears status bit, and value of 0 leaves bit intact. Bits are logged regardless of interrupt mask status. **Value** **Description** 0 INACTIVE 1 ACTIVE	RW	0x0
7	dcrc	Writes to bits clear status bit. Value of 1 clears status bit, and value of 0 leaves bit intact. Bits are logged regardless of interrupt mask status. **Value** **Description** 0 INACTIVE 1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
6	rcrc	Writes to bits clear status bit. Value of 1 clears status bit, and value of 0 leaves bit intact. Bits are logged regardless of interrupt mask status. **Value** **Description** 0 INACTIVE 1 ACTIVE	RW	0x0
5	rxdr	Writes to bits clear status bit. Value of 1 clears status bit, and value of 0 leaves bit intact. Bits are logged regardless of interrupt mask status. **Value** **Description** 0 INACTIVE 1 ACTIVE	RW	0x0
4	txdr	Writes to bits clear status bit. Value of 1 clears status bit, and value of 0 leaves bit intact. Bits are logged regardless of interrupt mask status. **Value** **Description** 0 INACTIVE 1 ACTIVE	RW	0x0
3	dto	Writes to bits clear status bit. Value of 1 clears status bit, and value of 0 leaves bit intact. Bits are logged regardless of interrupt mask status. **Value** **Description** 0 INACTIVE 1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
2	cmd	Writes to bits clear status bit. Value of 1 clears status bit, and value of 0 leaves bit intact. Bits are logged regardless of interrupt mask status. Value: Description 0: INACTIVE 1: ACTIVE	RW	0x0
1	re	Writes to bits clear status bit. Value of 1 clears status bit, and value of 0 leaves bit intact. Bits are logged regardless of interrupt mask status. Value: Description 0: INACTIVE 1: ACTIVE	RW	0x0
0	cd	Writes to bits clear status bit. Value of 1 clears status bit, and value of 0 leaves bit intact. Bits are logged regardless of interrupt mask status. Value: Description 0: INACTIVE 1: ACTIVE	RW	0x0

status

Status Register

Module Instance	Base Address	Register Address
i_sdmmc_sdmmc	0xFF808000	0xFF808048

Offset: 0x48

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dma_req RO 0x0	dma_ack RO 0x0	fifo_count RO 0x0													response_index RO 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
response_index RO 0x0					data_state_mc_busy RO 0x0	data_busy RO 0x0	data_3_status RO 0x1	command_fsm_states RO 0x0				fifo_full RO 0x0	fifo_empty RO 0x1	fifo_tx_watermark RO 0x1	fifo_rx_watermark RO 0x0

status Fields

Bit	Name	Description	Access	Reset
31	dma_req	DMA request signal state; either dw_dma_req or ge_dma_req, depending on DW-DMA or Generic-DMA selection.	RO	0x0
30	dma_ack	DMA acknowledge signal state; either dw_dma_ack or ge_dma_ack, depending on DW-DMA or Generic-DMA selection.	RO	0x0
29:17	fifo_count	FIFO count Number of filled locations in FIFO	RO	0x0
16:11	response_index	Index of previous response, including any auto-stop sent by core	RO	0x0
10	data_state_mc_busy	Data transmit or receive state-machine is busy Value: Description 0: DATASTATENOTBSY 1: DATASTATEBSY	RO	0x0

Bit	Name	Description	Access	Reset
9	data_busy	Inverted version of raw selected card_data[0] 0-card data not busy 1-card data busy **Value** **Description** 0 CARDNOTBUSY 1 CARDBUSY	RO	0x0
8	data_3_status	Raw selected card_data[3]; checks whether card is present 0-card not present 1-card present **Value** **Description** 0 CARDNOTPRESENT 1 CARDPRESENT	RO	0x1
7:4	command_fsm_states	Command FSM states: 0 Idle 1 Send init sequence 2 Tx cmd start bit 3 Tx cmd tx bit 4 Tx cmd index + arg 5 Tx cmd crc7 6 Tx cmd end bit 7 Rx resp start bit 8 Rx resp IRQ response 9 Rx resp tx bit 10 Rx resp cmd idx 11 Rx resp data	RO	0x0

Bit	Name	Description	Access	Reset
		12 Rx resp crc7 13 Rx resp end bit 14 Cmd path wait NCC 15 Wait; CMD-to-response turnaround NOTE: The command FSM state is represented using 19 bits. The STATUS Register(7:4) has 4 bits to represent the command FSM states. Using these 4 bits, only 16 states can be represented. Thus three states cannot be represented in the STATUS(7:4) register. The three states that are not represented in the STATUS Register(7:4) are: * Bit 16 Wait for CCS * Bit 17 Send CCSD * Bit 18 Boot Mode Due to this, while command FSM is in “Wait for CCS state” or “Send CCSD” or “Boot Mode”, the Status register indicates status as 0 for the bit field 7:4. **Value — Description** 0 — IDLEANDOTHERS 1 — SENDINITSEQ 2 — TXCMDSTART 3 — TXCMDTXBIT 4 — TXCMDINDXARG 5 — TXCMDCRC7 6 — TXCMDEND 7 — RXRESPSTART 8 — RXRESPIRQ 9 — RXRESPTX 10 — RXRESPCMDIDX 11 — RXRESPDATA 12 — RXRESPCRC7 13 — RXRESPEND 14 — CMDPATHWAIT		

Bit	Name	Description	Access	Reset
		Value Description 15 WAITCMDTURN		
3	fifo_full	FIFO is full status Value Description 0 FIFONOTFULL 1 FIFOFULL	RO	0x0
2	fifo_empty	FIFO is empty status Value Description 0 FIFONOTEMPTY 1 FIFOEMPTY	RO	0x1
1	fifo_tx_watermark	FIFO reached Transmit watermark level; not qualified with data transfer. Value Description 0 NOTXWATERMARK 1 TXWATERMARK	RO	0x1
0	fifo_rx_watermark	FIFO reached Receive watermark level; not qualified with data transfer. Value Description 0 RXWATERMARK 1 NORXWATERMARK	RO	0x0

fifoth

FIFO Threshold Watermark Register

Module Instance	Base Address	Register Address
i_sdmmc_sdmmc	0xFF808000	0xFF80804C

Offset: 0x4C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	dw_dma_multiple_transaction_size RW 0x0			rx_wmark RW 0x3FF											
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				tx_wmark RW 0x0											

fifoth Fields

Bit	Name	Description	Access	Reset
30:28	dw_dma_multiple_transaction_size	Burst size of multiple transaction; should be programmed same as DW-DMA controller multiple-transaction-size SRC/DEST_MSIZE. 000 1 transfers 001 4 010 8 011 16 100 32 101 64 110 128 111 256 The units for transfers is the H_DATA_WIDTH parameter. A single transfer (dw_dma_single assertion in case of Non DW DMA interface) would be signalled based on this value. Value should be sub-multiple of (RX_WMark + 1)* (F_DATA_WIDTH/H_DATA_WIDTH) and (FIFO_DEPTH - TX_WMark)* (F_DATA_WIDTH/H_DATA_WIDTH)	RW	0x0

Bit	Name	Description	Access	Reset
		For example, if FIFO_DEPTH = 16, FDATA_WIDTH == H_DATA_WIDTH Allowed combinations for MSize and TX_WMark are: MSize = 1, TX_WMARK = 1-15 MSize = 4, TX_WMark = 8 MSize = 4, TX_WMark = 4 MSize = 4, TX_WMark = 12 MSize = 8, TX_WMark = 8 MSize = 8, TX_WMark = 4 Allowed combinations for MSize and RX_WMark are: MSize = 1, RX_WMARK = 0-14 MSize = 4, RX_WMark = 3 MSize = 4, RX_WMark = 7 MSize = 4, RX_WMark = 11 MSize = 8, RX_WMark = 7 Recommended: MSize = 8, TX_WMark = 8, RX_WMark = 7 **Value** **Description** 0 TXMSIZE1 1 TXMSIZE4 2 TXMSIZEK8 3 TXMSIZEK16 5 RXMSIZEK1 6 RXMSIZEK4 7 RXMSIZE8		

Bit	Name	Description	Access	Reset
27:16	rx_wmark	FIFO threshold watermark level when receiving data to card. When FIFO data count reaches greater than this number,DMA/FIFO request is raised. During end of packet, request is generated regardless of threshold programming in order to complete any remaining data. In non-DMA mode, when receiver FIFO threshold (RXDR) interrupt is enabled, then interrupt is generated instead of DMA request. During end of packet, interrupt is not generated if threshold programming is larger than any remaining data. It is responsibility of host to read remaining bytes on seeing Data Transfer Done interrupt. In DMA mode, at end of packet, even if remaining bytes are less than threshold, DMA request does single transfers to flush out any remaining bytes before Data Transfer Done interrupt is set. 12 bits-1 bit less than FIFO-count of status register, which is 13 bits. Limitation: RX_WMark <= FIFO_DEPTH-2 Recommended: (FIFO_DEPTH/2) - 1; (means greater than (FIFO_DEPTH/2) - 1) NOTE: In DMA mode during CCS time-out, the DMA does not generate the request at the end of packet, even if remaining bytes are less than threshold. In this case, there will be some data left in the FIFO. It is the responsibility of the application to reset the FIFO after the CCS timeout.	RW	0x3FF

Bit	Name	Description	Access	Reset
11:0	tx_wmark	FIFO threshold watermark level when transmitting data to card. When FIFO data count is less than or equal to this number,DMA/FIFO request is raised. If Interrupt is enabled, then interrupt occurs. During end of packet, request or interrupt is generated,regardless of threshold programming. In non-DMA mode, when transmit FIFO threshold (TXDR) interrupt is enabled, then interrupt is generated instead of DMA request. During end of packet, on last interrupt, host is responsible for filling FIFO with only required remaining bytes (not before FIFO is full or after CIU completes data transfers, because FIFO may not be empty). In DMA mode, at end of packet, if last transfer is less than burst size, DMA controller does single cycles until required bytes are transferred. 12 bits-1 bit less than FIFO-count of status register, which is 13 bits. Limitation: TX_WMark >= 1; Recommended: FIFO_DEPTH/2; (means less than or equal to FIFO_DEPTH/2)	RW	0x0

cdetect

Card Detect Register

Module Instance	Base Address	Register Address
i_sdmmc_sdmmc	0xFF808000	0xFF808050

Offset: 0x50

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															card_detect_n RO 0x1

cdetect Fields

Bit	Name	Description	Access	Reset
0	card_detect_n	Value on card_detect_n input ports (1 bit per card); read-only bits.0 represents presence of card. Only NUM_CARDS number of bits are implemented. Value: Description 0: DETECTED 1: NOTDETECTED	RO	0x1

wrtprt

Card Detect Register

Module Instance	Base Address	Register Address
i_sdmmc_sdmmc	0xFF808000	0xFF808054

Offset: 0x54

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															write_protect RO 0x1

wrtprt Fields

Bit	Name	Description	Access	Reset
0	write_protect	Value on card_write_prt input ports (1 bit per card). 1 represents write protection. Only NUM_CARDS number of bits are implemented. **Value** **Description** 0 DISABLED 1 ENABLED	RO	0x1

tcbcnt

Transferred CIU Card Byte Count Register

Module Instance	Base Address	Register Address
i_sdmmc_sdmmc	0xFF808000	0xFF80805C

Offset: 0x5C

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
trans_card_byte_count RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
trans_card_byte_count RO 0x0															

tcbcnt Fields

Bit	Name	Description	Access	Reset
31:0	trans_card_byte_count	Number of bytes transferred by CIU unit to card. In 32-bit or 64-bit AMBA data-bus-width modes, register should be accessed in full to avoid read-coherency problems.In 16-bit AMBA data-bus-width mode, internal 16-bit coherency register is implemented. User should first read lower 16 bits and then higher 16 bits. When reading lower 16 bits, higher 16 bits of counter are stored in temporary register. When higher 16 bits are read, data from temporary register is supplied. Both TCBCNT and TBBCNT share same coherency register. When AREA_OPTIMIZED parameter is 1, register should be read only after data transfer completes; during data transfer,register returns 0.	RO	0x0

tbbcnt

Transferred Host to BIU-FIFO Byte Count Register

Module Instance	Base Address	Register Address
i_sdmmc_sdmmc	0xFF808000	0xFF808060

Offset: 0x60

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
trans_fifo_byte_count RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
trans_fifo_byte_count RO 0x0															

tbbcnt Fields

Bit	Name	Description	Access	Reset
31:0	trans_fifo_byte_count	Number of bytes transferred between Host/DMA memory and BIU FIFO. In 32-bit or 64-bit AMBA data-bus-width modes, register should be accessed in full to avoid read-coherency problems.In 16-bit AMBA data-bus-width mode, internal 16-bit coherency register is implemented. User should first read lower 16 bits and then higher 16 bits. When reading lower 16 bits, higher 16 bits of counter are stored in temporary register. When higher 16 bits are read, data from temporary register is supplied. Both TCBCNT and TBBCNT share same coherency register.	RO	0x0

debnce

Debounce Count Register

Module Instance	Base Address	Register Address
i_sdmmc_sdmmc	0xFF808000	0xFF808064

Offset: 0x64

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								debounce_count RW 0xFFFFFF							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
debounce_count RW 0xFFFFFF															

debnce Fields

Bit	Name	Description	Access	Reset
23:0	debounce_count	Number of host clocks (clk) used by debounce filter logic; typical debounce time is 5-25 ms.	RW	0xFFFFFF

usrid

User ID Register

Module Instance	Base Address	Register Address
i_sdmmc_sdmmc	0xFF808000	0xFF808068

Offset: 0x68

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
usr_id RW 0x7967797															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
usr_id RW 0x7967797															

usrid Fields

Bit	Name	Description	Access	Reset
31:0	usr_id	User identification register; value set by user. Default reset value can be picked by user while configuring core before synthesis. Can also be used as scratch pad register by user.	RW	0x7967797

verid

Version ID Register

Module Instance	Base Address	Register Address
i_sdmmc_sdmmc	0xFF808000	0xFF80806C

Offset: 0x6C

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ver_id RO 0x5342270A															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ver_id RO 0x5342270A															

verid Fields

Bit	Name	Description	Access	Reset
31:0	ver_id	Synopsys version identification register; register value is hard-wired.Can be read by firmware to support different versions of core.	RO	0x5342270A

hcon

Hardware Configuration Register

Module Instance	Base Address	Register Address
i_sdmmc_sdmmc	0xFF808000	0xFF808070

Offset: 0x70

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				ac RO 0x0	aro RO 0x0	ncd RO 0x0		scfp RO 0x1	ihr RO 0x1	rios RO 0x0	dmadatawidth RO 0x1			dmaintf RO 0x0	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
haddrwidth RO 0xC						hdatawidth RO 0x1			hbus RO 0x0	nc RO 0x0					ct RO 0x1

hcon Fields

Bit	Name	Description	Access	Reset
27	ac	For 64-bit Address Configuration Only - bit 27 0 - 32 bit addressing supported 1 - 64 bit address supported	RO	0x0
26	aro	Area optimized **Value** **Description** 0 NOTOPTFORAREA	RO	0x0
25:24	ncd	Number of clock dividers less one **Value** **Description** 0 ONEDIV	RO	0x0
23	scfp	Clock False Path **Value** **Description** 1 SET	RO	0x1
22	ihr	Implement hold register **Value** **Description** 1 IMPLEMENTED	RO	0x1
21	rios	FIFO RAM location **Value** **Description** 0 OUTSIDE	RO	0x0

Bit	Name	Description	Access	Reset
20:18	dmadatawidth	Encodes bit width of external DMA controller interface. Doesn't apply to the SD/MMC because it has no external DMA controller interface. **Value** **Description** 1 WIDTH32BITS	RO	0x1
17:16	dmaintf	DMA interface type **Value** **Description** 0 NONE	RO	0x0
15:10	haddrwidth	Slave bus address width less one **Value** **Description** 12 WIDTH13BITS	RO	0xC
9:7	hdatawidth	Slave bus data width **Value** **Description** 1 WIDTH32BITS	RO	0x1
6	hbus	Slave bus type. **Value** **Description** 0 APB	RO	0x0
5:1	nc	Maximum number of cards less one **Value** **Description** 0 NUMCARD	RO	0x0
0	ct	Supported card types **Value** **Description** 1 SDMMC	RO	0x1

uhs_reg

UHS-1 Register

Module Instance	Base Address	Register Address
i_sdmmc_sdmmc	0xFF808000	0xFF808074

Offset: 0x74

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ddr_reg RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
volt_reg RW 0x0															

uhs_reg Fields

Bit	Name	Description	Access	Reset
31:16	ddr_reg	DDR mode. These bits indicate DDR mode of operation to the core for the data transfer. 0 Non-DDR mode 1 DDR mode UHS_REG [16] should be set for card number 0, UHS_REG [17] for card number 1 and so on. **Value** **Description** 0 NONDDR 1 DDR	RW	0x0

Bit	Name	Description	Access	Reset
15:0	volt_reg	High Voltage mode. Determines the voltage fed to the buffers by an external voltage regulator. 0 Buffers supplied with 3.3V Vdd 1 Buffers supplied with 1.8V Vdd These bits function as the output of the host controller and are fed to an external voltage regulator. The voltage regulator must switch the voltage of the buffers of a particular card to either 3.3V or 1.8V, depending on the value programmed in the register. VOLT_REG[0] should be set to 1'b1 for card number 0 in order to make it operate for 1.8V. **Value** **Description** 0 BUF33V 1 BUF18V	RW	0x0

rst_n

Hardware Reset Register

Module Instance	Base Address	Register Address
i_sdmmc_sdmmc	0xFF808000	0xFF808078

Offset: 0x78

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															card_reset RW 0x1

rst_n Fields

Bit	Name	Description	Access	Reset
0	card_reset	Hardware reset. 1 Active mode 0 Reset These bits cause the cards to enter pre-idle state, which requires them to be re-initialized. ■CARD_RESET[0] should be set to 1'b0 to reset card number 0 ■CARD_RESET[15] should be set to 1'b0 to reset card number 15. The number of bits implemented is restricted to NUM_CARDS. **Value** **Description** 0 DEASSERT 1 ASSERT	RW	0x1

bmod

Bus Mode Register

Module Instance	Base Address	Register Address
i_sdmmc_sdmmc	0xFF808000	0xFF808080

Offset: 0x80

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					pbl RO 0x0			de RW 0x0	dsl RW 0x0					fb RW 0x0	swr RW 0x0

bmod Fields

Bit	Name	Description	Access	Reset
10:8	pbl	Programmable Burst Length. These bits indicate the maximum number of beats to be performed in one IDMAC transaction. The IDMAC will always attempt to burst as specified in PBL each time it starts a Burst transfer on the host bus. The permissible values are 1, 4, 8, 16, 32, 64, 128 and 256. This value is the mirror of MSIZE of FIFOTH register. In order to change this value, write the required value to FIFOTH register. This is an encode value as follows. 000 1 transfers 001 4 transfers 010 8 transfers 011 16 transfers 100 32 transfers 101 64 transfers 110 128 transfers 111 256 transfers Transfer unit is either 16, 32, or 64 bits, based on HDATA_WIDTH. PBL is a read-only value and is applicable only for Data Access; it does not apply to descriptor accesses. **Value — Description** 0 — TRANS1 1 — TRANS4 2 — TRANS8 3 — TRANS16 4 — TRANS32 5 — TRANS64 6 — TRANS128 7 — TRANS256	RO	0x0

Bit	Name	Description	Access	Reset
7	de	IDMAC Enable. When set, the IDMAC is enabled. DE is read/write. **Value** **Description** 0 DISABLED 1 ENABLED	RW	0x0
6:2	dsl	Descriptor Skip Length. Specifies the number of HWord/Word/Dword (depending on 16/32/64-bit bus) to skip between two unchained descriptors. This is applicable only for dual buffer structure. DSL is read/write.	RW	0x0
1	fb	Fixed Burst. Controls whether the AHB Master interface performs fixed burst transfers or not. When set,the AHB will use only SINGLE, INCR4, INCR8 or INCR16 during start of normal burst transfers.When reset,the AHB will use SINGLE and INCR burst transfer operations. FB is read/write. **Value** **Description** 0 NOFIXEDBRST 1 FIXEDBRST	RW	0x0
0	swr	Software Reset.When set,the DMA Controller resets all its internal registers. SWR is read/write. It is automatically cleared after 1 clock cycle. **Value** **Description** 0 NOSFTRESET 1 SFTRESET	RW	0x0

pldmnd

Poll Demand Register

Module Instance	Base Address	Register Address
i_sdmmc_sdmmc	0xFF808000	0xFF808084

Offset: 0x84

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
pd WO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
pd WO 0x0															

pldmnd Fields

Bit	Name	Description	Access	Reset
31:0	pd	Poll Demand. If the OWN bit of a descriptor is not set, the FSM goes to the Suspend state. The host needs to write any value into this register for the IDMAC FSM to resume normal descriptor fetch operation. This is a write only register. PD bit is write-only.	WO	0x0

dbaddr

Descriptor List Base Address Register

Module Instance	Base Address	Register Address
i_sdmmc_sdmmc	0xFF808000	0xFF808088

Offset: 0x88

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
sdl RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
sdl RW 0x0															

dbaddr Fields

Bit	Name	Description	Access	Reset
31:0	sdl	Start of Descriptor List. Contains the base address of the First Descriptor. The LSB bits [0/1/2:0] for 16/32/64-bit bus-width) are ignored and taken as all-zero by the IDMAC internally. Hence these LSB bits are read-only.	RW	0x0

idsts

Internal DMAC Status Register

Module Instance	Base Address	Register Address
i_sdmmc_sdmmc	0xFF808000	0xFF80808C

Offset: 0x8C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															fsm RO 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
fsm RO 0x0			eb RO 0x0			ais RW 0x0	nis RW 0x0	Reserved		ces RW 0x0	du RW 0x0	Reserved	fbe RW 0x0	ri RW 0x0	ti RW 0x0

idsts Fields

Bit	Name	Description	Access	Reset
16:13	fsm	DMAC FSM present state. 0 DMA_IDLE 1 DMA_SUSPEND 2 DESC_RD 3 DESC_CHK 4 DMA_RD_REQ_WAIT 5 DMA_WR_REQ_WAIT 6 DMA_RD 7 DMA_WR 8 DESC_CLOSE This bit is read-only. **Value** **Description** 0 DMAIDLE 1 DMASUSPEND 2 DESCRD 3 DESCCHK 4 DMARDREQWAIT 5 DMAWRREQWAIT 6 DMARD 7 DMAWR 8 DECCLOSE	RO	0x0

Bit	Name	Description	Access	Reset
12:10	eb	Error Bits. Indicates the type of error that caused a Bus Error. Valid only with Fatal Bus Error bitIDSTS[2] set. This field does not generate an interrupt. 3'b001 Host Abort received during transmission 3'b010 Host Abort received during reception Others: Reserved EB is read-only. **Value** **Description** 1 HOSTARBTTX 2 HOSTARBRX	RO	0x0
9	ais	Abnormal Interrupt Summary. Logical OR of the following: ■IDSTS[2]-Fatal Bus Interrupt ■IDSTS[4]-DU bit Interrupt Only unmasked bits affect this bit. This is a sticky bit and must be cleared each time a corresponding bit that causes AIS to be set is cleared. Writing a 1 clears this bit. **Value** **Description** 0 NOCLR 1 CLR	RW	0x0

Bit	Name	Description	Access	Reset
8	nis	Normal Interrupt Summary. Logical OR of the following: ■IDSTS[0]-Transmit Interrupt ■IDSTS[1]-Receive Interrupt Only unmasked bits affect this bit. This is a sticky bit and must be cleared each time a corresponding bit that causes NIS to be set is cleared. Writing a 1 clears this bit. **Value** **Description** 0 NOCLR 1 CLR	RW	0x0

Bit	Name	Description	Access	Reset
5	ces	Card Error Summary. Indicates the status of the transaction to/from the card; also present in RINTSTS. Indicates the logical OR of the following bits: ■EBE End Bit Error ■RTO Response Timeout/ Boot Ack Timeout ■RCRC Response CRC ■SBE Start Bit Error ■DRTO Data Read Timeout/BDS timeout ■DCRC Data CRC for Receive ■RE Response Error Writing a 1 clears this bit. The abort condition of the IDMAC depends on the setting of this CES bit. If the CES bit is enabled, then the IDMAC aborts on a “response error”; however, it will not abort if the CES bit is cleared. **Value** **Description** 0 NOCLR 1 CLR	RW	0x0
4	du	Descriptor Unavailable Interrupt. This bit is set when the descriptor is unavailable due to OWN bit = 0 (DES0[31] =0). Writing a 1 clears this bit. **Value** **Description** 0 NOCLR 1 CLR	RW	0x0

Bit	Name	Description	Access	Reset
2	fbe	Fatal Bus Error Interrupt. Indicates that a Bus Error occurred (IDSTS[12:10]). When this bit is set, the DMA disables all its bus accesses. Writing a 1 clears this bit. **Value** — **Description** 0 — NOCLR 1 — CLR	RW	0x0
1	ri	Receive Interrupt. Indicates the completion of data reception for a descriptor. Writing a 1 clears this bit. **Value** — **Description** 0 — NOCLR 1 — CLR	RW	0x0
0	ti	Transmit Interrupt. Indicates that data transmission is finished for a descriptor. Writing a '1' clears this bit. **Value** — **Description** 0 — NOCLR 1 — CLR	RW	0x0

idinten

Internal DMAC Interrupt Enable Register

Module Instance	Base Address	Register Address
i_sdmmc_sdmmc	0xFF808000	0xFF808090

Offset: 0x90

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						ai RW 0x0	ni RW 0x0	Reserved		ces RW 0x0	du RW 0x0	Reserved	fbe RW 0x0	ri RW 0x0	ti RW 0x0

idinten Fields

Bit	Name	Description	Access	Reset
9	ai	Abnormal Interrupt Summary Enable. When set, an abnormal interrupt is enabled. This bit enables the following bits: ■IDINTEN[2] - Fatal Bus Error Interrupt ■IDINTEN[4] - DU Interrupt **Value** **Description** 0 DISABLED 1 ENABLED	RW	0x0
8	ni	Normal Interrupt Summary Enable. When set, a normal interrupt is enabled. When reset, a normal interrupt is disabled. This bit enables the following bits: ■IDINTEN[0] - Transmit Interrupt ■IDINTEN[1] - Receive Interrupt **Value** **Description** 0 DISABLED 1 ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
5	ces	Card Error summary Interrupt Enable. When set, it enables the Card Interrupt summary. **Value** **Description** 0 DISABLED 1 ENABLED	RW	0x0
4	du	Descriptor Unavailable Interrupt. When set along with Abnormal Interrupt Summary Enable, the DU interrupt is enabled. **Value** **Description** 0 DISABLED 1 ENABLED	RW	0x0
2	fbe	Fatal Bus Error Enable. When set with Abnormal Interrupt Summary Enable, the Fatal Bus Error Interrupt is enabled. When reset, Fatal Bus Error Enable Interrupt is disabled. **Value** **Description** 0 DISABLED 1 ENABLED	RW	0x0
1	ri	Receive Interrupt Enable. When set with Normal Interrupt Summary Enable, Receive Interrupt is enabled. When reset, Receive Interrupt is disabled. **Value** **Description** 0 DISABLED 1 ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
0	ti	Transmit Interrupt Enable. When set with Normal Interrupt Summary Enable, Transmit Interrupt is enabled. When reset, Transmit Interrupt is disabled. Value: Description 0: DISABLED 1: ENABLED	RW	0x0

dscaddr

Current Host Descriptor Address Register

Module Instance	Base Address	Register Address
i_sdmmc_sdmmc	0xFF808000	0xFF808094

Offset: 0x94

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hda RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
hda RO 0x0															

dscaddr Fields

Bit	Name	Description	Access	Reset
31:0	hda	Host Descriptor Address Pointer. Cleared on reset. Pointer updated by IDMAC during operation. This register points to the start address of the current descriptor read by the IDMAC.	RO	0x0

bufaddr

Current Buffer Descriptor Address Register

Module Instance	Base Address	Register Address
i_sdmmc_sdmmc	0xFF808000	0xFF808098

Offset: 0x98

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hba RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
hba RO 0x0															

bufaddr Fields

Bit	Name	Description	Access	Reset
31:0	hba	Host Buffer Address Pointer. Cleared on Reset. Pointer updated by IDMAC during operation. This register points to the current Data Buffer Address being accessed by the IDMAC.	RO	0x0

cardthrctl

Card Threshold Control Register

Module Instance	Base Address	Register Address
i_sdmmc_sdmmc	0xFF808000	0xFF808100

Offset: 0x100

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved			cardrdthreshold RW 0x0												
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														busy_clr_int_en RW 0x0	cardrdthren RW 0x0

cardthrctl Fields

Bit	Name	Description	Access	Reset
28:16	cardrdthreshold	Card Read Threshold size; N depends on the FIFO size: ■N = 27 FIFO_DEPTH is 128 ■N = 26 FIFO_DEPTH is 64 ■N = 25 FIFO_DEPTH is 32 ■N = 24 FIFO_DEPTH is 16 ■N = 23 FIFO_DEPTH is 8 Note: The maximum programmable value of Card Read Threshold size is 512.	RW	0x0

Bit	Name	Description	Access	Reset
1	busy_clr_int_en	Busy Clear Interrupt generation: ■1'b0 - Busy Clear Interrupt disabled ■1'b1 - Busy Clear Interrupt enabled Note: The application can disable this feature if it does not want to wait for a Busy Clear Interrupt. For example, in a multi-card scenario, the application can switch to the other card without waiting for a busy to be completed. In such cases, the application can use the polling method to determine the status of busy. By default this feature is disabled and backward-compatible to the legacy drivers where polling is used.	RW	0x0
0	cardrdthren	Card Read Threshold Enable ■1'b0 - Card Read Threshold disabled ■1'b1 - Card Read Threshold enabled. Host Controller initiates Read Transfer only if CardRdThreshold amount of space is available in receive FIFO. **Value** **Description** 0 DISABLED 1 ENABLED	RW	0x0

back_end_power_r

Back End Power Register

Module Instance	Base Address	Register Address
i_sdmmc_sdmmc	0xFF808000	0xFF808104

Offset: 0x104

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
back_end_power RW 0x0															

back_end_power_r Fields

Bit	Name	Description	Access	Reset
15:0	back_end_power	Back end power ■1'b0 Off; Reset ■1'b1 Back-end Power supplied to card application; one pin per card **Value** **Description** 0 BACKEND0 1 BACKEND1	RW	0x0

data

Provides read/write access to data FIFO. Addresses 0x200 and above are mapped to the data FIFO. More than one address is mapped to data FIFO so that FIFO can be accessed using bursts.

Module Instance	Base Address	Register Address
i_sdmmc_sdmmc	0xFF808000	0xFF808200

Offset: 0x200

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
value RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
value RW 0x0															

data Fields

Bit	Name	Description	Access	Reset
31:0	value	Provides read/write access to data FIFO.	RW	0x0

gpio

General Purpose Input/Output Register

Module Instance	Base Address	Register Address
i_sdmmc_sdmmc	0xFF808000	0xFF808058

Offset: 0x58

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								gpo RW 0x0							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
gpo RW 0x0								gpi RO 0x0							

gpio Fields

Bit	Name	Description	Access	Reset
23:8	gpo	Value needed to be driven to gpo pins; this portion of register is read/write. Valid only when AREA_OPTIMIZED parameter is 0.	RW	0x0
7:0	gpi	Value on gpi input ports; this portion of register is read-only. Valid only when AREA_OPTIMIZED parameter is 0.	RO	0x0

uhs_reg_ext

UHS Register Extention

Module Instance	Base Address	Register Address
i_sdmmc_sdmmc	0xFF808000	0xFF808108

Offset: 0x108

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ext_clk_mux_ctrl RW 0x0		clk_drv_phase_ctrl RW 0x0							clk_smpl_phase_ctrl RW 0x0						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
mmc_volt_reg RW 0x0															

uhs_reg_ext Fields

Bit	Name	Description	Access	Reset
31:30	ext_clk_mux_ctrl	Input clock control for cclk_in. The MUX controlled by these bits exists outside DWC_mobile_storage IP.	RW	0x0
29:23	clk_drv_phase_ctrl	Control for amount of phase shift on cclk_in_drv clock. Can choose three MSBs to control delay lines and four LSBs to control phase shift; alternatively, use only LSBs.	RW	0x0
22:16	clk_smpl_phase_ctrl	Control for amount of phase shift on cclk_in_sample clock. Can choose three MSBs to control delay lines and four LSBs to control phase shift; alternatively, use only LSBs.	RW	0x0
15:0	mmc_volt_reg	Support for 1.2V. MMC_VOLT_REG bits; must be read in combination with UHS_VOLT_REG to decode output selected voltage. The biu_volt_reg_1_2[NUM_CARD_BUS-1:0] signal decodes the voltage combination selected for the I/O voltage logic. Host controllers that support only SD standard or standard versions before eMMC4.41 do not program MMC_VOLT_REG. Only host controllers that support all three versions3.3,1.8,1.2 Vcan program MMC_VOLT_REG and connect biu_volt_reg_1_2.	RW	0x0

emmc_ddr_reg

EMMC DDR Register

Module Instance	Base Address	Register Address
i_sdmmc_sdmmc	0xFF808000	0xFF80810C

Offset: 0x10C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															half_start_bit RW 0x0

emmc_ddr_reg Fields

Bit	Name	Description	Access	Reset
0	half_start_bit	Control for start bit detection mechanism inside DWC_mobile_storage based on duration of start bit; each bit refers to one slot. For eMMC 4.5, start bit can be: ■Full cycle (HALF_START_BIT = 0) ■Less than one full cycle (HALF_START_BIT = 1) Set HALF_START_BIT=1 for eMMC 4.5 and above; set to 0 for SD applications.	RW	0x0

enable_shift

Register to control the amount of shift on enables

Module Instance	Base Address	Register Address
i_sdmmc_sdmmc	0xFF808000	0xFF808110

Offset: 0x110

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														enable_shift_card RW 0x0	

enable_shift Fields

Bit	Name	Description	Access	Reset
1:0	enable_shift_card	Control for the amount of phase shift provided on the default enables in the design. Two bits are assigned for each card/slot. For example, bits[1:0] control slot0 and indicate the following. ■00 Default phase shift ■01 Enables shifted to next immediate positive edge ■10 Enables shifted to next immediate negative edge ■11 Reserved	RW	0x0

sdmmc_ecc Address Map

Module Instance	Base Address	End Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2FFF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
IP_REV_ID on page 11-215	0x0	32	RO	0x0	
CTRL on page 11-215	0x8	32	RW	0x0	ECC Control Register
INITSTAT on page 11-216	0xC	32	RW	0x0	This bit is used to set the initialize the memory and ecc to a known value
ERRINTEN on page 11-217	0x10	32	RW	0x0	Error Interrupt enable
ERRINTENS on page 11-218	0x14	32	RW	0x0	Error Interrupt set
ERRINTENR on page 11-219	0x18	32	RW	0x0	Error Interrupt reset.
INTMODE on page 11-220	0x1C	32	RW	0x0	Reads reflect SERRINTEN.
INTSTAT on page 11-220	0x20	32	RW	0x0	This bit is used to enable interrupt generation on SERR lookup table overflow. When all the entries in the table are valid=1 and this is bit is enabled, serr_req signal will be asserted.
INTTEST on page 11-221	0x24	32	RW	0x0	This bits is used to test interrupt from ECC RAM to GIC
MODSTAT on page 11-222	0x28	32	RW	0x0	Counter feature status flag
DERRADDRA on page 11-223	0x2C	32	RO	0x0	This register shows the address of PORTA current double-bit error. RAM size will determine the maximum number of address bits.

Register	Offset	Width	Access	Reset Value	Description
SERRADDRA on page 11-224	0x30	32	RO	0x0	This register shows the address of PORTA current single-bit error. RAM size will determine the maximum number of address bits.
DERRADDRB on page 11-224	0x34	32	RO	0x0	This register shows the address of PORTB current double-bit error. RAM size will determine the maximum number of address bits.
SERRADDRB on page 11-225	0x38	32	RO	0x0	This register shows the address of PORTB current single-bit error. RAM size will determine the maximum number of address bits.
SERRCNTREG on page 11-226	0x3C	32	RW	0x0	Maximum counter value for single-bit error interrupt
ECC_Addrbus on page 11-227	0x40	32	RW	0x0	MSB bit of address is determined by ADR.
ECC_RData0bus on page 11-227	0x44	32	RO	0x0	Data will be read to this register field.
ECC_RData1bus on page 11-228	0x48	32	RO	0x0	Data will be read to this register field.
ECC_RData2bus on page 11-229	0x4C	32	RO	0x0	Data will be read to this register field.
ECC_RData3bus on page 11-229	0x50	32	RO	0x0	Data will be read to this register field.
ECC_WData0bus on page 11-230	0x54	32	WO	0x0	Data from the register will be written to the RAM.
ECC_WData1bus on page 11-231	0x58	32	WO	0x0	Data from the register will be written to the RAM.

Register	Offset	Width	Access	Reset Value	Description
ECC_WData2bus on page 11-231	0x5C	32	WO	0x0	Data from the register will be written to the RAM.
ECC_WData3bus on page 11-232	0x60	32	WO	0x0	Data from the register will be written to the RAM.
ECC_RDataecc0bus on page 11-233	0x64	32	RO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.
ECC_RDataecc1bus on page 11-233	0x68	32	RO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.
ECC_WDataecc0bus on page 11-234	0x6C	32	WO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.
ECC_WDataecc1bus on page 11-235	0x70	32	WO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.
ECC_dbytectrl on page 11-236	0x74	32	RW	0x0	Max number of implemented byte enabled is DAT/8
ECC_accctrl on page 11-237	0x78	32	RW	0x0	These bits determine which byte of data/ecc to write to RAM.

Register	Offset	Width	Access	Reset Value	Description
ECC_startacc on page 11-238	0x7C	32	RW	0x0	These bits determine which byte of data/ecc to write to RAM.
ECC_wdctrl on page 11-239	0x80	32	RW	0x0	Bits to Enable/Disable Watch Dog Timer
SERRLKUPA0 on page 11-239	0x90	32	RW	0x0	Single-bit error address in LOOKUP TABLE for PORTA.
SERRLKUPB0 on page 11-240	0xD0	32	RW	0x0	Single-bit error address in LOOKUP TABLE for PORTB.

sdmmc_ecc Summary

Base Address: 0xFF8C2C00

Register Address Offset	Bit Fields															
ecc_sdmmc_ecc_register-Block																
IP_REV_ID 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SIREV RO 0x0															
CTRL 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved							INITB 0x0	Reserved							INITA 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						CNT_RSTB 0x0	CNT_RSTA 0x0	Reserved							ECC_EN 0x0

Register Address Offset	Bit Fields															
INITSTAT 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							INITCOMPLETEB 0x0	Reserved							INITCOM-PLETEA 0x0
ERRINTEN 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTEN 0x0
ERRINTENS 0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTS 0x0
ERRINTENR 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTR 0x0
INTMODE 0x1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															INTONCMP 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							INTONOVF 0x0	Reserved							INTMODE 0x0
INTSTAT 0x20	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved							DERRPENB 0x0	Reserved							SERRPENB 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							DERRPENA 0x0	Reserved							SERRPENA 0x0

Register Address Offset	Bit Fields															
INTTEST 0x24	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved							TDERRB 0x0	Reserved							TSERRB 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							TDERRA 0x0	Reserved							TSERRA 0x0
MODSTAT 0x28	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														CMPFLGB 0x0	CMPFLGA 0x0
DERRADDRA 0x2C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						Address 0x0									
SERRADDRA 0x30	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						Address 0x0									
DERRADDRB 0x34	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						Address 0x0									
SERRADDRB 0x38	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						Address 0x0									

Register Address Offset	Bit Fields															
SERRCNTREG 0x3C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	SERRCNT 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SERRCNT 0x0															
ECC_Addrbus 0x40	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						ECC_AddrBUS 0x0									
ECC_RData0bus 0x44	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_RData1bus 0x48	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_RData2bus 0x4C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_RData3bus 0x50	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															

Register Address Offset	Bit Fields															
ECC_WData0bus 0x54	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_WData1bus 0x58	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_WData2bus 0x5C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_WData3bus 0x60	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_RDataecc0bus 0x64	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_RDataecc3BUS 0x0							Reserved	ECC_RDataecc2BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_RDataecc1BUS 0x0							Reserved	ECC_RDataecc0BUS 0x0						
ECC_RDataecc1bus 0x68	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_RDataecc7BUS 0x0							Reserved	ECC_RDataecc6BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_RDataecc5BUS 0x0							Reserved	ECC_RDataecc4BUS 0x0						

Register Address Offset	Bit Fields															
ECC_WDataecc0bus 0x6C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_WDataecc3BUS 0x0							Reserved	ECC_WDataecc2BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_WDataecc1BUS 0x0							Reserved	ECC_WDataecc0BUS 0x0						
ECC_WDataecc1bus 0x70	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_WDataecc7BUS 0x0							Reserved	ECC_WDataecc6BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_WDataecc5BUS 0x0							Reserved	ECC_WDataecc4BUS 0x0						
ECC_dbytectrl 0x74	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															DBEN 0x0
ECC_accctrl 0x78	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							RDWR 0x0	Reserved						ECCOVR 0x0	DATAOVR 0x0
ECC_startacc 0x7C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															ENBUSA 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															ENBUSB 0x0
ECC_wdctrl 0x80	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															WDEN_RAM 0x0

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SERRLKUPA0 0x90	VALID RW 0x0	Reserved														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						Address RO 0x0									
SERRLKUPB0 0xD0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	VALID RW 0x0	Reserved														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						Address RO 0x0									

IP_REV_ID

Module Instance	Base Address	Register Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2C00

Offset: 0x0

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SIREV RO 0x0															

IP_REV_ID Fields

Bit	Name	Description	Access	Reset
15:0	SIREV	IP Rev # These bits indicate the silicon revision number.	RO	0x0

CTRL

ECC Control Register

Module Instance	Base Address	Register Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2C08

Offset: 0x8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							INITB 0x0	Reserved							INITA 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						CNT_RSTB 0x0	CNT_RSTA 0x0	Reserved							ECC_EN 0x0

CTRL Fields

Bit	Name	Description	Access	Reset
24	INITB	Enable for the hardware memory initialization PORTB.	RW	0x0
16	INITA	Enable for the hardware memory initialization PORTA.	RW	0x0
9	CNT_RSTB	Enable to reset internal single-bit error counter B value to zero	RW	0x0

Bit	Name	Description	Access	Reset
8	CNT_RSTA	Enable to reset internal single-bit error counter A value to zero	RW	0x0
0	ECC_EN	Enable for the ECC detection and correction logic.	RW	0x0

INITSTAT

This bit is used to set the initialize the memory and ecc to a known value

Module Instance	Base Address	Register Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2C0C

Offset: 0xC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							INITCOMPLETEB 0x0	Reserved							INITCOMPLETEA 0x0

INITSTAT Fields

Bit	Name	Description	Access	Reset
8	INITCOMPLETEB	This bit is used to verify if the hardware memory initialization has completed PORTB.	RW	0x0
0	INITCOMPLETEA	This bit is used to verify if the hardware memory initialization has completed PORTB.	RW	0x0

ERRINTEN

Error Interrupt enable

Module Instance	Base Address	Register Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2C10

Offset: 0x10

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															SERRINTEN 0x0

ERRINTEN Fields

Bit	Name	Description	Access	Reset
0	SERRINTEN	This bit is used to enable the single bit error interrupt of ECC RAM system	RW	0x0

ERRINTENS

Error Interrupt set

Module Instance	Base Address	Register Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2C14

Offset: 0x14

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															SERRINTS 0x0

ERRINTENS Fields

Bit	Name	Description	Access	Reset
0	SERRINTS	This bit is used to set the single-bit error interrupt bit.	RW	0x0

ERRINTENR

Error Interrupt reset.

Module Instance	Base Address	Register Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2C18

Offset: 0x18

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															SERRINTR 0x0

ERRINTENR Fields

Bit	Name	Description	Access	Reset
0	SERRINTR	This bit is used to reset the single-bit error interrupt bit. Reads reflect SERRINTEN. 1'b0: Writing of zero has no effect. 1'b1: By writing one, this bit will reset SERRINTEN bit to 0. This is performing a bitwise writing of this feature.	RW	0x0

INTMODE

Reads reflect SERRINTEN.

Module Instance	Base Address	Register Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2C1C

Offset: 0x1C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															INTONCMP 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							INTONOVF 0x0	Reserved							INTMODE 0x0

INTMODE Fields

Bit	Name	Description	Access	Reset
16	INTONCMP	Enable interrupt on compare.	RW	0x0
8	INTONOVF	Enable interrupt on overflow.	RW	0x0

Bit	Name	Description	Access	Reset
0	INTMODE	Interrupt mode for single-bit errors.	RW	0x0

INTSTAT

This bit is used to enable interrupt generation on SERR lookup table overflow. When all the entries in the table are valid=1 and this is bit is enabled, serr_req signal will be asserted.

Module Instance	Base Address	Register Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2C20

Offset: 0x20

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							DERRPENB 0x0	Reserved							SERRPENB 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							DERRPENA 0x0	Reserved							SERRPENA 0x0

INTSTAT Fields

Bit	Name	Description	Access	Reset
24	DERRPENB	Double-bit error pending PORTB.	RW	0x0
16	SERRPENB	Single-bit error pending for PORTB.	RW	0x0
8	DERRPENA	Double-bit error pending for PORTA.	RW	0x0
0	SERRPENA	Single-bit error pending for PORTA.	RW	0x0

INTTEST

This bits is used to test interrupt from ECC RAM to GIC

Module Instance	Base Address	Register Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2C24

Offset: 0x24

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							TDERRB 0x0	Reserved							TSERRB 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							TDERRA 0x0	Reserved							TSERRA 0x0

INTTEST Fields

Bit	Name	Description	Access	Reset
24	TDERRB	Test PORTB Double-bit error.	RW	0x0
16	TSERRB	Test PORTB Single-bit error.	RW	0x0
8	TDERRA	Test PORTA Double-bit error.	RW	0x0
0	TSERRA	Test PORTA Single-bit error.	RW	0x0

MODSTAT

Counter feature status flag

Module Instance	Base Address	Register Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2C28

Offset: 0x28

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														CMPFLGB 0x0	CMPFLGA 0x0

MODSTAT Fields

Bit	Name	Description	Access	Reset
1	CMPFLGB	Port B compare status flag	RW	0x0
0	CMPFLGA	Port A compare status flag	RW	0x0

DERRADDRA

This register shows the address of PORTA current double-bit error. RAM size will determine the maximum number of address bits.

Module Instance	Base Address	Register Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2C2C

Offset: 0x2C

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						Address 0x0									

DERRADDRA Fields

Bit	Name	Description	Access	Reset
9:0	Address	Recent double-bit error address.	RO	0x0

SERRADDRA

This register shows the address of PORTA current single-bit error. RAM size will determine the maximum number of address bits.

Module Instance	Base Address	Register Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2C30

Offset: 0x30

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						Address 0x0									

SERRADDRA Fields

Bit	Name	Description	Access	Reset
9:0	Address	Recent single-bit error address.	RO	0x0

DERRADDRB

This register shows the address of PORTB current double-bit error. RAM size will determine the maximum number of address bits.

Module Instance	Base Address	Register Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2C34

Offset: 0x34

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						Address 0x0									

DERRADDRB Fields

Bit	Name	Description	Access	Reset
9:0	Address	Recent double-bit error address.	RO	0x0

SERRADDRB

This register shows the address of PORTB current single-bit error. RAM size will determine the maximum number of address bits.

Module Instance	Base Address	Register Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2C38

Offset: 0x38

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						Address 0x0									

SERRADDRB Fields

Bit	Name	Description	Access	Reset
9:0	Address	Recent single-bit error address.	RO	0x0

SERRCNTREG

Maximum counter value for single-bit error interrupt

Module Instance	Base Address	Register Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2C3C

Offset: 0x3C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SERRCNT 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SERRCNT 0x0															

SERRCNTREG Fields

Bit	Name	Description	Access	Reset
31:0	SERRCNT	Counter value	RW	0x0

ECC_Addrbus

MSB bit of address is determined by ADR.

Module Instance	Base Address	Register Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2C40

Offset: 0x40

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						ECC_AddrBUS 0x0									

ECC_Addrbus Fields

Bit	Name	Description	Access	Reset
9:0	ECC_AddrBUS	Address will be driven to RAM to either read or write the data. Address will be latched by the RAM when the Enbus is asserted.	RW	0x0

ECC_RData0bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2C44

Offset: 0x44

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData0bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_RDataBUS	ECC_RDataBUS[31:0].	RO	0x0

ECC_RData1bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2C48

Offset: 0x48

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData1bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_RDataBUS	ECC_RDataBUS[63:32].	RO	0x0

ECC_RData2bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2C4C

Offset: 0x4C

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData2bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_RDataBUS	ECC_RDataBUS[95:64].	RO	0x0

ECC_RData3bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2C50

Offset: 0x50

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData3bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_RDataBUS	ECC_RDataBUS[127-96].	RO	0x0

ECC_WData0bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2C54

Offset: 0x54

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData0bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_WDataBUS	ECC_WDataBUS[31:0].	WO	0x0

ECC_WData1bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2C58

Offset: 0x58

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData1bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_WDataBUS	ECC_WDataBUS[63:32].	WO	0x0

ECC_WData2bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2C5C

Offset: 0x5C

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData2bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_WDataBUS	ECC_WDataBUS[95-64].	WO	0x0

ECC_WData3bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2C60

Offset: 0x60

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData3bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_WDataBUS	ECC_WDataBUS[127-96].	WO	0x0

ECC_RDataecc0bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2C64

Offset: 0x64

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	ECC_RDataecc3BUS 0x0							Reserved	ECC_RDataecc2BUS 0x0						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	ECC_RDataecc1BUS 0x0							Reserved	ECC_RDataecc0BUS 0x0						

ECC_RDataecc0bus Fields

Bit	Name	Description	Access	Reset
30:24	ECC_RDataecc3BUS	Eccdata will be read to this register field.	RO	0x0
22:16	ECC_RDataecc2BUS	Eccdata will be read to this register field.	RO	0x0
14:8	ECC_RDataecc1BUS	Eccdata will be read to this register field.	RO	0x0
6:0	ECC_RDataecc0BUS	Eccdata will be read to this register field.	RO	0x0

ECC_RDataecc1bus

The msb bit for the register is configured based on DAT

parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2C68

Offset: 0x68

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	ECC_RDataecc7BUS 0x0							Reserved	ECC_RDataecc6BUS 0x0						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	ECC_RDataecc5BUS 0x0							Reserved	ECC_RDataecc4BUS 0x0						

ECC_RDataecc1bus Fields

Bit	Name	Description	Access	Reset
30:24	ECC_RDataecc7BUS	Eccdata will be read to this register field.	RO	0x0
22:16	ECC_RDataecc6BUS	Eccdata will be read to this register field.	RO	0x0
14:8	ECC_RDataecc5BUS	Eccdata will be read to this register field.	RO	0x0
6:0	ECC_RDataecc4BUS	Eccdata will be read to this register field.	RO	0x0

ECC_WDataecc0bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2C6C

Offset: 0x6C

Access: WO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	ECC_WDataecc3BUS 0x0							Reserved	ECC_WDataecc2BUS 0x0						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	ECC_WDataecc1BUS 0x0							Reserved	ECC_WDataecc0BUS 0x0						

ECC_WDataecc0bus Fields

Bit	Name	Description	Access	Reset
30:24	ECC_WDataecc3BUS	Eccdata from the register will be written to the RAM.	WO	0x0
22:16	ECC_WDataecc2BUS	Eccdata from the register will be written to the RAM.	WO	0x0
14:8	ECC_WDataecc1BUS	Eccdata from the register will be written to the RAM.	WO	0x0
6:0	ECC_WDataecc0BUS	Eccdata from the register will be written to the RAM.	WO	0x0

ECC_WDataecc1bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2C70

Offset: 0x70

Access: WO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	ECC_WDataecc7BUS 0x0							Reserved	ECC_WDataecc6BUS 0x0						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	ECC_WDataecc5BUS 0x0							Reserved	ECC_WDataecc4BUS 0x0						

ECC_WDataecc1bus Fields

Bit	Name	Description	Access	Reset
30:24	ECC_WDataecc7BUS	Eccdata from the register will be written to the RAM.	WO	0x0
22:16	ECC_WDataecc6BUS	Eccdata from the register will be written to the RAM.	WO	0x0
14:8	ECC_WDataecc5BUS	Eccdata from the register will be written to the RAM.	WO	0x0
6:0	ECC_WDataecc4BUS	Eccdata from the register will be written to the RAM.	WO	0x0

ECC_dbytectrl

ECC Data Byte Control. The maximum number of implemented byte enables equals the data width of the RAM divided by 8.

Module Instance	Base Address	Register Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2C74

Offset: 0x74

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															DBEN 0x0

ECC_dbytectrl Fields

Bit	Name	Description	Access	Reset
0	DBEN	Byte or word enable for access.	RW	0x0

ECC_accctrl

These bits determine which byte of data/ecc to write to RAM.

Module Instance	Base Address	Register Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2C78

Offset: 0x78

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							RDWR 0x0	Reserved						ECCOVR 0x0	DATAOVR 0x0

ECC_accctrl Fields

Bit	Name	Description	Access	Reset
8	RDWR	Control for read/write.	RW	0x0
1	ECCOVR	ECC Data Override.	RW	0x0
0	DATAOVR	RAM Data Override. Override the ECC_dataBUS register with RAM data in read mode set by ECC_RW. 1'b0: Data override disabled. 1'b1: Data override enabled.	RW	0x0

ECC_startacc

These bits determine which byte of data/ecc to write to RAM.

Module Instance	Base Address	Register Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2C7C

Offset: 0x7C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															ENBUSA 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															ENBUSB 0x0

ECC_startacc Fields

Bit	Name	Description	Access	Reset
16	ENBUSA	Start RAM access for PORTA.	RW	0x0
0	ENBUSB	Start RAM access for PORTB.	RW	0x0

ECC_wdctrl

Bits to Enable/Disable Watch Dog Timer

Module Instance	Base Address	Register Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2C80

Offset: 0x80

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															WDEN_RAM 0x0

ECC_wdctrl Fields

Bit	Name	Description	Access	Reset
0	WDEN_RAM	Enable watchdog timeout for OCP register access to IP RAM.	RW	0x0

SERRLKUPA0

Single-bit error address in LOOKUP TABLE for PORTA.

Module Instance	Base Address	Register Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2C90

Offset: 0x90

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VALID RW 0x0	Reserved														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						Address RO 0x0									

SERRLKUPA0 Fields

Bit	Name	Description	Access	Reset
31	VALID	Valid flag bit. Valid bit indicates if the address in this register is current or stale.	RW	0x0
9:0	Address	Recent Single-bit error address. This register shows the address of the each single-bit error. RAM size will determine the maximum number of address bits. If ram size is 32 Kbytes, bit 30-16 will be reserved and read as zero.	RO	0x0

SERRLKUPB0

Single-bit error address in LOOKUP TABLE for PORTB.

Module Instance	Base Address	Register Address
ecc_sdmmc_ecc_registerBlock	0xFF8C2C00	0xFF8C2CD0

Offset: 0xD0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VALID RW 0x0	Reserved														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						Address RO 0x0									

SERRLKUPB0 Fields

Bit	Name	Description	Access	Reset
31	VALID	Valid flag bit. Valid bit indicates if the address in this register is current or stale.	RW	0x0
9:0	Address	Recent Single-bit error address. This register shows the address of the each single-bit error. RAM size will determine the maximum number of address bits. If ram size is 32 Kbytes, bit 30-16 will be reserved and read as zero.	RO	0x0

Document Revision History

Table 14-38: Document Revision History

Date	Version	Changes
October 2016	2016.10.28	Removed SPI support in tables in the Features section.
May 2016	2016.05.27	Added a link to the *Supported Flash Devices for Arria 10 SoC* webpage.
May 2016	2016.05.03	Maintenance release
November 2015	2015.11.02	• Moved "Interface Signals" section below "SD/MMC Controller Block Diagram and System Integration" section and renamed to "SD/MMC Signal Description." Clarified signals in this section. • Removed the indication that the AV/CV HPS support 8-bit eMMC. • Added information that Card Detect is only supported on interfaces routed via the FPGA fabric.
May 2015	2015.05.04	Added information about clearing out the ECC before the feature is enabled
December 2014	2014.12.15	Maintenance release
August 2014	2014.08.18	Initial release

Quad SPI Flash Controller 15

2016.10.28

a10_5v4 Subscribe Send Feedback

The hard processor system (HPS) provides a quad serial peripheral interface (SPI) flash controller for access to serial NOR flash devices. The quad SPI flash controller supports standard SPI flash devices as well as high-performance dual and quad SPI flash devices. The quad SPI flash controller is based on Cadence Quad SPI Flash Controller (QSPI_FLASH_CTRL).

Features of the Quad SPI Flash Controller

The quad SPI flash controller supports the following features:

- SPIx1, SPIx2, or SPIx4 (quad SPI) serial NOR flash devices
- Any device clock frequencies up to 108 MHz(44)
- Direct access and indirect access modes
- Single I/O, dual I/O, or quad I/O instructions
- Up to four chip selects
- DMA transfers using the HPS DMA controller
- Configurable clock polarity and phase
- Programmable write-protected regions
- Programmable delays between transactions
- Programmable device sizes
- Read data capture tuning
- Local buffering with ECC logic for indirect transfers
- Support eXecute-In-Place(XIP) mode
- Supports the Micron N25Q00AA11GSF40F (1024 Mb, 108 MHz) Quad SPI flash memory that is verified working with the HPS

Related Information

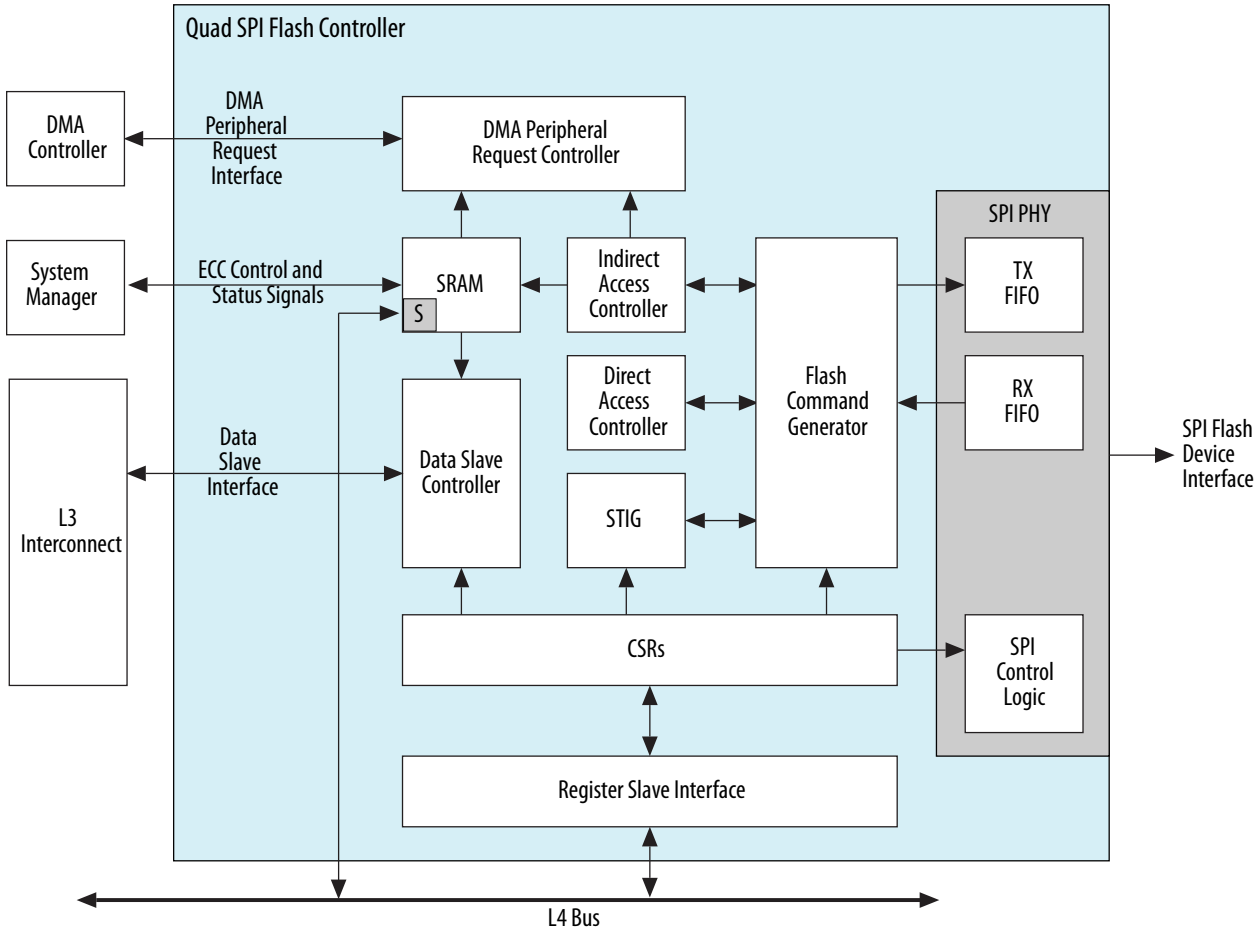
Supported Flash Devices for Arria 10 SoC

For more information, refer to the supported QSPI flash devices section on this page.

(44) The quad SPI controller supports any device frequencies, but this speed is limited by the supported flash devices.

Quad SPI Flash Controller Block Diagram and System Integration

Figure 15-1: Quad SPI Flash Controller Block Diagram and System Integration



The quad SPI controller consists of the following blocks and interfaces:

- Register slave interface - Provides access to the control and status registers (CSRs)
- Data slave controller - Interface and controller that provides the following functionality:
  - Performs data transfers to and from the level 3 (L3) interconnect
  - Validates incoming accesses
  - Performs byte or half-word reordering
  - Performs write protection
  - Forwards transfer requests to direct and indirect controller
- Direct access controller - provides memory-mapped slaves direct access to the flash memory
- Indirect access controller - provides higher-performance access to the flash memory through local buffering and software transfer requests
- Software triggered instruction generator (STIG) - generates flash commands through the flash command register (`flashcmd`) and provides low-level access to flash memory

- Flash command generator - generates flash command and address instructions based on instructions from the direct and indirect access controllers or the STIG
- DMA peripheral request controller - issues requests to the DMA peripheral request interface to communicate with the HPS DMA controller
- SPI PHY - serially transfers data and commands to the external SPI flash devices

The static RAM (SRAM) connected to the indirect access controller, DMA peripheral request controller, and data slave controller has an error correction code (ECC) controller built-in to provide ECC protection. The ECC controller is able to detect single-bit and double-bit errors, and correct the single-bit errors. The ECC operation and functionality is programmable via the ECC register slave interface, as shown in **Figure 15-1**. The ECC register interface provides host access to configure the ECC logic as well as inject bit errors into the memory. It also provides host access to memory initialization hardware used to clear out the memory contents including the ECC bits. The ECC controller generates interrupts upon occurrences of single and double-bit errors, and the interrupt signals are connected to the system manager.

Quad SPI Flash Controller Signal Description

The quad SPI controller provides four chip select outputs to allow control of up to four external quad SPI flash devices. The outputs serve different purposes depending on whether the device is used in single, dual, or quad operation mode. The following table lists the I/O pin use of the quad SPI controller interface signals for each operation mode. For more information on which signals are available to the FPGA and HPS I/O, refer to the *HPS Component Interfaces* chapter.

Table 15-1: Interface Pins

Signal	Pin	Mode	Direction	Function
`qspi_io0_i`	`IO0`	Single	Output	Data output 0
`qspi_io0_o` `qspi_mo_oe[0]`		Dual or quad	Bidirectional	Data I/O 0
`qspi_io1_i`	`IO1`	Single	Input	Data input 0
`qspi_io1_o` `qspi_mo_oe[1]`		Dual or quad	Bidirectional	Data I/O 1
`qspi_io2_i`	`IO2_WPN`	Single or dual	Output	Active low write protect
`qspi_io2_wpn_o` `qspi_mo_oe[2]`		Quad	Bidirectional	Data I/O 2
`qspi_io3_i`	`IO3_HOLD`	Single, dual, or quad	Bidirectional	Data I/O 3
`qspi_io3_hold_o` `qspi_mo_oe[3]`				

Signal	Pin	Mode	Direction	Function
qspi_ss_o[0]	SS0	Single, dual, or quad	Output	Active low slave select 0
qspi_ss_o[1]	SS1			Active low slave select 1
qspi_ss_o[2]	SS2			Active low slave select 2
qspi_ss_o[3]	SS3			Active low slave select 3
qspi_sclk_out	CLK	Single	Output	QSPI serial clock output which connects to the FPGA core fabric
qspi_s2f_clk				QSPI serial clock output which connects to the FPGA clock network **Note:** Use the qspi_s2f_clk as the serial clock output to an external SPI device.

Related Information

HPS Component Interfaces on page 28-8
For more information on routing Quad SPI Flash Controller interface signals to the FPGA and HPS I/O, refer to this chapter.

Functional Description of the Quad SPI Flash Controller

Overview

The quad SPI flash controller uses the register slave interface to select the operation modes and configure the data slave interface for data transfers. The quad SPI flash controller uses the data slave interface for direct and indirect accesses, and the register slave interface for software triggered instruction generator (STIG) operation and SPI legacy mode accesses.

Accesses to the data slave are forwarded to the direct or indirect access controller. If the access address is within the configured indirect address range, the access is sent to the indirect access controller.

Data Slave Interface

The quad SPI flash controller uses the data slave interface for direct, indirect, and SPI legacy mode accesses.

The data slave interface is 32 bits wide and permits byte, half-word, and word accesses. For write accesses, incrementing burst lengths of 1, 4, 8 and 16 are supported. For read accesses, all burst types and sizes are supported.

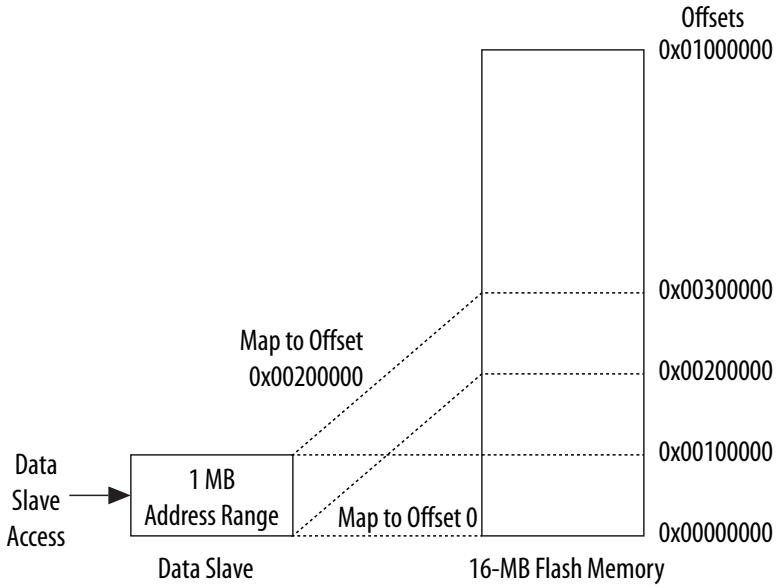
Direct Access Mode

In direct access mode, an access to the data slave triggers a read or write command to the flash memory. To use the direct access mode, enable the direct access controller with the enable direct access controller bit (`endiracc`) of the quad SPI configuration register (`cfg`).

An external master, for example a processor, triggers the direct access controller with a read or write operation to the data slave interface. The data slave exposes a 1 MB window into the flash device. You can remap this window to any 1 MB location within the flash device.

Data Slave Remapping Example

Figure 15-2: Data Slave Remapping Example



To remap the data slave to access other 1 MB regions of the flash device, enable address remapping in the enable ARM® AMBA® advanced high speed bus (AHB) address remapping field (`enahbremap`) of the `cfg` register. All incoming data slave accesses remap to the offset specified in the remap address register (`remapaddr`).

The 20 LSBs of incoming addresses are used for accessing the 1 MB region and the higher bits are ignored.

Note: The quad SPI controller does not issue any error status for accesses that lie outside the connected flash memory space.

AHB

The data slave interface is throttled as the read or write burst is carried out. The latency is designed to be as small as possible and is kept to a minimum when the use of XIP read instructions are enabled.

FLASH erase operations, which may be required before a page write, are triggered by software using the documented programming interface. They are not issued automatically.

Once a page program cycle has been started, the QSPI Flash Controller will automatically poll for the write cycle to complete before allowing any further data slave interface accesses to complete. This is achieved by holding any subsequent AHB direct accesses in wait state.

Indirect Access Mode

In indirect access mode, flash data is temporarily buffered in the quad SPI controller's static RAM (SRAM). Software controls and triggers indirect accesses through the register slave interface. The controller transfers data through the data slave interface.

Indirect Read Operation

An indirect read operation reads data from the flash memory, places the data into the SRAM, and transfers the data to an external master through the data slave interface. The indirect read operations are controlled by the following registers:

- Indirect read transfer register (`indrd`)
- Indirect read transfer watermark register (`indrdwater`)
- Indirect read transfer start address register (`indrdstaddr`)
- Indirect read transfer number bytes register (`indrdcnt`)
- Indirect address trigger register (`indaddrtrig`)

These registers need to be configured prior to issuing indirect read operations. The start address needs to be defined in the `indrdstaddr` register and the total number of bytes to be fetched is specified in the `indircnt` register. Writing 1 to the start indirect read bit (`start`) of the `indrd` register triggers the indirect read operation from the flash memory to populate the SRAM with the returned data.

To read data from the flash device into the SRAM, an external master issues 32-bit read transactions to the data slave interface. The address of the read access must be in the indirect address range. You can configure the indirect address through the `indaddrtrig` register. The external master can issue 32-bit reads until the last word of an indirect transfer. On the final read, the external master may issue a 32-bit, 16-bit or 8-bit read to complete the transfer. If there are less than four bytes of data to read on the last transfer, the external master can still issue a 32-bit read and the quad SPI controller will pad the upper bits of the response data with zeros.

Assuming the requested data is present in the SRAM at the time the data slave read is received by the quad SPI controller, the data is fetched from SRAM and the response to the read burst is achieved with minimum latency. If the requested data is not immediately present in the SRAM, the data slave interface enters a wait state until the data has been read from flash memory into SRAM. Once the data has been read from SRAM by the external master, the quad SPI controller frees up the associated resource in the SRAM. If the SRAM is full, reads on the SPI interface are backpressured until space is available in the SRAM. The quad SPI controller completes any current read burst, waits for SRAM to free up, and issues a new read burst at the address where the previous burst was terminated.

The processor can also use the SRAM fill level in the SRAM fill register (`sramfill`) to control when data should be fetched from the SRAM.

Alternatively, you can configure the fill level watermark of the SRAM in the `indrdwater` register. When the SRAM fill level passes the watermark level, the indirect transfer watermark interrupt is generated. You can disable this watermark feature by writing a value of all zeroes to the `indrdwater` register.

For the final bytes of data read by the quad SPI controller and placed in the SRAM, if the watermark level is greater than zero, the indirect transfer watermark interrupt is generated even when the actual SRAM fill level has not risen above the watermark.

If the address of the read access is outside the range of the indirect trigger address, one of the following actions occurs:

- When direct access mode is enabled, the read uses direct access mode.
- When direct access mode is disabled, the slave returns an error back to the requesting master.

You can cancel an indirect operation by setting the cancel indirect read bit (`cancel`) of the `indrd` register to 1. For more information, refer to the "Indirect Read Operation with DMA Disabled" section.

Related Information

Indirect Read Operation with DMA Disabled on page 15-17

Indirect Write Operation

An indirect write operation programs data from the SRAM to the flash memory. The indirect write operations are controlled by the following registers:

- Indirect write transfer register (`indwr`)
- Indirect write transfer watermark register (`indwrwater`)
- Indirect write transfer start address register (`indwrstaddr`)
- Indirect write transfer number bytes register (`indwrcnt`)
- `indaddrtrig` register

These registers need to be configured prior to issuing indirect write operations. The start address needs to be defined in the `indwrstaddr` register and the total number of bytes to be written is specified in the `indwrcnt` register. The start indirect write bit (`start`) of the `indwr` register triggers the indirect write operation from the SRAM to the flash memory.

To write data from the SRAM to the flash device, an external master issues 32-bit write transactions to the data slave. The address of the write access must be in the indirect address range. You can configure the indirect address through the `indaddrtrig` register. The external master can issue 32-bit writes until the last word of an indirect transfer. On the final write, the external master may issue a 32-bit, 16-bit or 8-bit write to complete the transfer. If there are less than four bytes of data to write on the last transfer, the external master can still issue a 32-bit write and the quad SPI controller discards the extra bytes.

The SRAM size can limit the amount of data that the quad SPI controller can accept from the external master. If the SRAM is not full at the point of the write access, the data is pushed to the SRAM with minimum latency. If the external master attempts to push more data to the SRAM than the SRAM can accept, the quad SPI controller backpressures the external master with wait states. When the SRAM resource is freed up by pushing the data from SRAM to the flash memory, the SRAM is ready to receive more data from the external master. When the SRAM holds an equal or greater number of bytes than the size of a flash page, or when the SRAM holds all the remaining bytes of the current indirect transfer, the quad SPI controller initiates a write operation to the flash memory.

The processor can also use the SRAM fill level, in the `sramfill` register, to control when to write more data into the SRAM.

Alternatively, you can configure the fill level watermark of the SRAM in the `indwrwater` register. When the SRAM fill level falls below the watermark level, an indirect transfer watermark interrupt is generated to tell the software to write the next page of data to the SRAM. Because the quad SPI controller initiates non-end-of-data writes to the flash memory only when the SRAM contains a full flash page of data, you must set the watermark level to a value greater than one flash page to avoid the system stalling. You can disable this watermark feature by writing a value of all ones to the `indwrwater` register.

If the address of the write access is outside the range of the indirect trigger address, one of the following actions occurs:

- When direct access mode is enabled, the write uses direct access mode.
- When direct access mode is disabled, the slave returns an error back to the requesting master.

You can cancel an indirect operation by setting the cancel indirect write bit (`cancel`) of the `indwr` register to 1. For more information, refer to the "Indirect Write Operation with DMA Disabled" section.

Related Information

Indirect Write Operation with DMA Disabled on page 15-18

Consecutive Reads and Writes

It is possible to trigger two indirect operations at a time by triggering the `start` bit of the `indrd` or `indwr` register twice in short succession. The second operation can be triggered while the first operation is in progress. For example, software may trigger an indirect read or write operation while an indirect write operation is in progress. The corresponding start and count registers must be configured properly before software triggers each transfer operation.

This approach allows for a short turnaround time between the completion of one indirect operation and the start of a second operation. Any attempt to queue more than two operations causes the indirect read reject interrupt to be generated.

SPI Legacy Mode

SPI legacy mode allows software to access the internal TX FIFO and RX FIFO buffers directly, thus bypassing the direct, indirect and STIG controllers. Software accesses the TX FIFO and RX FIFO buffers by writing any value to any address through the data slave while legacy mode is enabled. You can enable legacy mode with the legacy IP mode enable bit (`enlegacyip`) of the `cfg` register.

Legacy mode allows the user to issue any flash instruction to the flash device, but imposes a heavy software overhead in order to manage the fill levels of the FIFO buffers effectively. The legacy SPI mode is bidirectional in nature, with data continuously being transferred both directions while the chip select is enabled. If the driver only needs to read data from the flash device, dummy data must be written to ensure the chip select stays active, and vice versa for write transactions.

For example, to perform a basic read of four bytes to a flash device that has three address bytes, software must write a total of eight bytes to the TX FIFO buffer. The first byte would be the instruction opcode, the next three bytes are the address, and the final four bytes would be dummy data to ensure the chip select stays active while the read data is returned. Similarly, because eight bytes were written to the TX FIFO buffer, software should expect eight bytes to be returned in the RX FIFO buffer. The first four bytes of this would be discarded, leaving the final four bytes holding the data read from the device.

Because the TX FIFO and RX FIFO buffers are four bytes deep each, software must maintain the FIFO buffer levels to ensure the TX FIFO buffer does not underflow and the RX FIFO buffer does not overflow. Interrupts are provided to indicate when the fill levels pass the watermark levels, which are configurable through the TX threshold register (`txtresh`) and RX threshold register (`rxtresh`).

Register Slave Interface

The quad SPI flash controller uses the register slave interface to configure the quad SPI controller through the quad SPI configuration registers, and to access flash memory under software control, through the `flashcmd` register in the STIG.

STIG Operation

The Software Triggered Instruction Generator (STIG) is used to access the volatile and non-volatile configuration registers, the legacy SPI status register, and other status and protection registers. The STIG also is used to perform ERASE functions. The direct and indirect access controllers are used only to

transfer data. The `flashcmd` register uses the following parameters to define the command to be issued to the flash device:

- Instruction opcode
- Number of address bytes
- Number of dummy bytes
- Number of write data bytes
- Write data
- Number of read data bytes

The address is specified through the flash command address register (`flashcmdaddr`). Once these settings have been specified, software can trigger the command with the execute command field (`execcmd`) of the `flashcmd` register and wait for its completion by polling the command execution status bit (`cmdexecstat`) of the `flashcmd` register. A maximum of eight data bytes may be read from the flash command read data lower (`flashcmdrddatalo`) and flash command read data upper (`flashcmdrddataup`) registers or written to the flash command write data lower (`flashcmdwrdatalo`) and flash command write data upper (`flashcmdwrdataup`) registers per command.

Commands issued through the STIG have a higher priority than all other read accesses and therefore interrupt any read commands being requested by the direct or indirect controllers. However, the STIG does not interrupt a write sequence that may have been issued through the direct or indirect access controller. In these cases, it might take a long time for the `cmdexecstat` bit of the `flashcmd` register indicates the operation is complete.

Note: Altera recommends using the STIG instead of the SPI legacy mode to access the flash device registers and perform erase operations.

Local Memory Buffer

The SRAM local memory buffer is a 128 by 32-bit (512 total bytes) memory and includes support for error correction code (ECC). The ECC controller is able to detect single-bit and double-bit errors, and correct the single-bit errors. The ECC operation and functionality is programmable through the ECC register slave interface, as shown in **Figure 15-1**. The ECC register interface provides host access to configure the ECC logic as well as inject bit errors into the memory. It also provides host access to memory initialization hardware used to clear out the memory contents including the ECC bits.

To prevent spurious ECC errors, software must use the memory initialization block in the ECC controller to initialize the entire memory data and ECC bits. The initialization block clears the memory data. Enabling initialization in the ECC Control Register is independent of enabling the ECC. The ECC controller generates interrupts upon occurrences of single- and double-bit errors. The interrupt requests are sent to the system manager which routes them to the General Interrupt Controller (GIC).

The SRAM has two partitions, with the lower partition reserved for indirect read operations and the upper partition for indirect write operations. The size of the partitions is specified in the SRAM partition register (`srampart`), based on 32-bit word sizes. For example, to specify four bytes of storage, write the value 1. The value written to the indirect read partition size field (`addr`) defines the number of entries reserved for indirect read operations. For example, write the value 32 (0x20) to partition the 128-entry SRAM to 32 entries (25%) for read usage and 96 entries (75%) for write usage.

Related Information

- **Memory Data Initialization** on page 11-5
  For more information about clearing memory data before enabling ECC, refer to "Memory Data Initialization" in the Error Checking and Correction section.

- **Error Checking and Correction Controller** on page 11-1
  For more information about ECC, refer to the *Error Checking and Correction Controller* chapter of the Arria 10 Device Handbook.

DMA Peripheral Request Controller

The DMA peripheral request controller is only used for the indirect mode of operation where data is temporarily stored in the SRAM. The quad SPI flash controller uses the DMA peripheral request interface to trigger the external DMA into performing data transfers between memory and the quad SPI controller.

There are two DMA peripheral request interfaces, one for indirect reads and one for indirect writes. The DMA peripheral request controller can issue two types of DMA requests, single or burst, to the external DMA. The number of bytes for each single or burst request is specified in the number of single bytes (`numsglreqbytes`) and number of burst bytes (`numburstreqbytes`) fields of the DMA peripheral register (`dmaper`). The DMA peripheral request controller splits the total amount of data to be transferred into a number of DMA burst and single requests by dividing the total number of bytes by the number of bytes specified in the burst request, and then dividing the remainder by the number of bytes in a single request.

Note: When programming the DMA controller, the burst request size must match the burst request size set in the quad SPI controller to avoid quickly reaching an overflow or underflow condition.

For indirect reads, the DMA peripheral request controller only issues DMA requests after the data has been retrieved from flash memory and written to SRAM. The rate at which DMA requests are issued depends on the watermark level. The `indrdwater` register defines the minimum fill level in bytes at which the DMA peripheral request controller can issue the DMA request. The higher this number is, the more data that must be buffered in SRAM before the external DMA moves the data. When the SRAM fill level passes the watermark level, the transfer watermark reached interrupt is generated.

For example, consider the following conditions:

- The total amount of data to be read using indirect mode is 256 bytes
- The SRAM watermark level is set at 128 bytes
- Software configures the burst type transfer size to 64 bytes

Under these conditions, the DMA peripheral request controller issues the first DMA burst request when the SRAM fill level passes 128 bytes (the watermark level). The DMA peripheral request controller triggers consecutive DMA burst requests as long as there is sufficient data in the SRAM to perform burst type requests. In this example, DMA peripheral request controller can issue at least two consecutive DMA burst requests to transfer a total of 128 bytes. If there is sufficient data in the SRAM, the DMA peripheral request controller requests the third DMA burst immediately. Otherwise the DMA peripheral request controller waits for the SRAM fill level to pass the watermark level again to trigger the next burst request. When the watermark level is triggered, there is sufficient data in the SRAM to perform the third and fourth burst requests to complete the entire transaction.

For the indirect writes, the DMA peripheral request controller issues DMA requests immediately after the transfer is triggered and continues to do so until the entire indirect write transfer has been transferred. The rate at which DMA requests are issued depends on the watermark level. The `indwrwater` register defines the maximum fill level in bytes at which the controller can issue the first DMA burst or single request. When the SRAM fill level falls below the watermark level, the transfer watermark reached interrupt is generated. When there is one flash page of data in the SRAM, the quad SPI controller initiates the write operation from SRAM to the flash memory.

Software can disable the DMA peripheral request interface with the `endma` field of the `cfg` register. If a master other than the DMA performs the data transfer for indirect operations, the DMA peripheral request interface must be disabled. By default, the indirect watermark registers are set to zero, which means the DMA peripheral request controller can issue DMA request as soon as possible.

Related Information

DMA Controller on page 16-1

Arbitration between Direct/Indirect Access Controller and STIG

When multiple controllers are active simultaneously, a fixed-priority arbitration scheme is used to arbitrate between each interface and access the external FLASH. The fixed priority is defined as follows, highest priority first.

1. The Indirect Access Write
2. The Direct Access Write
3. The STIG
4. The Direct Access Read
5. The Indirect Access Read

Each controller is back pressured while waiting to be serviced.

Configuring the Flash Device

For read and write accesses, software must initialize the device read instruction register (`devrd`) and the device write instruction register (`devwr`). These registers include fields to initialize the instruction opcodes that should be used as well as the instruction type, and whether the instruction uses single, dual or quad pins for address and data transfer. To ensure the quad SPI controller can operate from a reset state, the opcode registers reset to opcodes compatible with single I/O flash devices.

The quad SPI flash controller uses the instruction transfer width field (`instwidth`) of the `devrd` register to set the instruction transfer width for both reads and writes. There is no `instwidth` field in the `devwr` register. If instruction type is set to dual or quad mode, the address transfer width (`addrwidth`) and data transfer width (`datawidth`) fields of both registers are redundant because the address and data type is based on the instruction type. Thus, software can support the less common flash instructions where the opcode, address, and data are sent on two or four lanes. For most instructions, the opcodes are sent serially to the flash device, even for dual and quad instructions. One of the flash devices that supports instructions that can send the opcode over two or four lanes is the Micron N25Q128. For reference, **Table 15-2** and **Table 15-3** show how software should configure the quad SPI controller for each specific read and write instruction, respectively, supported by the Micron N25Q128 device.

Table 15-2: Quad SPI Configuration for Micron N25Q128 Device (Read Instructions)

Instruction	Lanes Used By Opcode	Lanes Used to Send Address	Lanes Used to Send Data	instwidth Value	addrwidth Value	datawidth Value
Read	1	1	1	0	0	0
Fast read	1	1	1	0	0	0
Dual output fast read (DOFR)	1	1	2	0	0	1
Dual I/O fast read (DIOFR)	1	2	2	0	1	1

Instruction	Lanes Used By Opcode	Lanes Used to Send Address	Lanes Used to Send Data	instwidth Value	addrwidth Value	datawidth Value
Quad output fast read (QOFR)	1	1	4	0	0	2
Quad I/O fast read (QIOFR)	1	4	4	0	2	2
Dual command fast read (DCFR)	2	2	2	1	Don't care	Don't care
Quad command fast read (QCFR)	4	4	4	2	Don't care	Don't care

Table 15-3: Quad SPI Configuration for Micron N25Q128 Device (Write Instructions)

Instruction	Lanes Used By Opcode	Lanes Used to Send Address	Lanes Used to Send Data	instwidth Value	addrwidth Value	datawidth Value
Page program	1	1	1	0	0	0
Dual input fast program (DIFP)	1	1	2	0	0	1
Dual input extended fast program (DIEFP)	1	2	2	0	1	1
Quad input fast program (QIFP)	1	1	4	0	0	2
Quad input extended fast program (QIEFP)	1	4	4	0	2	2
Dual command fast program (DCFP)	2	2	2	1	Don't care	Don't care
Quad command fast program (QCFP)	4	4	4	2	Don't care	Don't care

XIP Mode

The quad SPI controller supports XIP mode, if the flash devices support XIP mode. Depending on the flash device, XIP mode puts the flash device in read-only mode, reducing command overhead.

The quad SPI controller must instruct the flash device to enter XIP mode by sending the mode bits. When the enter XIP mode on next read bit (`enterxipnextrd`) of the `cfg` register is set to 1, the quad SPI controller and the flash device are ready to enter XIP mode on the next read instruction. When the enter XIP mode immediately bit (`enterxipimm`) of the `cfg` register is set to 1, the quad SPI controller and flash device enter XIP mode immediately.

When the `enterxipnextrd` or `enterxipimm` bit of the `cfg` register is set to 0, the quad SPI controller and flash device exit XIP mode on the next read instruction. For more information, refer to the *"XIP Mode Operations"* section.

Related Information

XIP Mode Operations on page 15-19

Write Protection

You can program the controller to write protect a specific region of the flash device. The protected region is defined as a set of blocks, specified by a starting and ending block. Writing to an area of protected flash region memory generates an error and triggers an interrupt.

You define the block size by specifying the number of bytes per block through the number of bytes per block field (`bytespersubsector`) of the device size register (`devsz`). The lower write protection register (`lowwrprot`) specifies the first flash block in the protected region. The upper write protection register (`uppwrprot`) specifies the last flash block in the protected region.

The write protection enable bit (`en`) of the write protection register (`wrprot`) enables and disables write protection. The write protection inversion bit (`inv`) of the `wrprot` register flips the definition of protection so that the region specified by `lowwrprt` and `uppwrprt` is unprotected and all flash memory outside that region is protected.

Data Slave Sequential Access Detection

The quad SPI flash controller detects sequential accesses to the data slave interface by comparing the current access with the previous access. An access is sequential when it meets the following conditions:

- The address of the current access sequentially follows the address of the previous access.
- The direction of the current access (read or write) is the same as previous access.
- The size of the current access (byte, half-word, or word) is the same as previous access.

When the access is detected as nonsequential, the sequential access to the flash device is terminated and a new sequential access begins. Altera recommends accessing the data slave sequentially. Sequential access has less command overhead, and therefore, increases data throughput.

Clocks

There are two clock inputs to the quad SPI controller: `l4_mp_clk` and `l4_main_clk`; and one clock output: `qspi_clk`. The quad SPI flash controller uses the `l4_mp_clk` clock to clock the data slave transfers and register slave accesses. The `l4_main_clk` clock is the reference clock for the quad SPI controller and is used to serialize the data and drive the external SPI interface. The `qspi_clk` clock is the generated clock source for the connected flash devices.

The following is true for the following reference clocks:

- l4_main_clk should be greater than l4_mp_clk
- l4_main_clk must be greater than two times qspi_clk

The qspi_clk clock is derived by dividing down the reference clock, l4_main_clk clock by the baud rate divisor field (bauddiv) of the cfg register.

Related Information

Clock Manager on page 2-1

Clock Gating

You can clock gate qspi_ref_clk through software. By programming the qspiclken bit of the en register in the perpllgrp you can enable or disable the qspi_ref_clk to the QSPI Controller.

Related Information

Clock Manager Address Map and Register Definitions on page 2-18

Resets

A single reset signal (qspi_flash_rst_n) is provided as an input to the quad SPI controller. The reset manager drives the reset signal to the ECC controller on a cold or warm reset. This resets the logic inside the ECC controller.

Related Information

Reset Manager on page 3-1

Taking the Quad SPI Flash Controller Out of Reset

When a cold or warm reset is issued in the HPS, the reset manager resets this module and holds it in reset until software releases it.

After the Cortex-A9 MPCore CPU boots, it can deassert the reset signal by clearing the appropriate bits in the reset manager's corresponding reset register. For details about reset registers, refer to "Module Reset Signals".

Interrupts

All interrupt sources are combined to create a single level-sensitive, active-high interrupt (qspi_intr). Software can determine the source of the interrupt by reading the interrupt status register (irqstat). By default, the interrupt source is cleared when software writes a one (1) to the interrupt status register. The interrupts are individually maskable through the interrupt mask register (irqmask). **Table 15-4** lists the interrupt sources in the irqstat register.

Table 15-4: Interrupt Sources in the irqstat Register

Interrupt Source	Description
Underflow detected	When 0, no underflow has been detected. When 1, the data slave write data is being supplied too slowly. This situation can occur when data slave write data is being supplied too slowly to keep up with the requested write operation This bit is reset only by a system reset and cleared only when a 1 is written to it.
Indirect operation complete	The controller has completed a triggered indirect operation.
Indirect read reject	An indirect operation was requested but could not be accepted because two indirect operations are already in the queue.
Protected area write attempt	A write to a protected area was attempted and rejected.
Illegal data slave access detected	An illegal data slave access has been detected. Data slave wrapping bursts and the use of split and retry accesses can cause this interrupt. It is usually an indicator that soft masters in the FPGA fabric are attempting to access the HPS in an unsupported way.
Transfer watermark reached	The indirect transfer watermark level has been reached.
Receive overflow	This condition occurs only in legacy SPI mode. When 0, no overflow has been detected. When 1, an over flow to the RX FIFO buffer has occurred. This bit is reset only by a system reset and cleared to zero only when this register is written to. If a new write to the RX FIFO buffer occurs at the same time as a register is read, this flag remains set to 1.
TX FIFO not full	This condition occurs only in legacy SPI mode. When 0, the TX FIFO buffer is full. When 1, the TX FIFO buffer is not full.
TX FIFO full	This condition occurs only in legacy SPI mode. When 0, the TX FIFO buffer is not full. When 1, the TX FIFO buffer is full.
RX FIFO not empty	This condition occurs only in legacy SPI mode. When 0, the RX FIFO buffer is empty. When 1, the RX FIFO buffer is not empty.

Interrupt Source	Description
RX FIFO full	This condition occurs only in legacy SPI mode. When 0, the RX FIFO buffer is not full. When 1, the RX FIFO buffer is full.
Indirect read partition overflow	Indirect Read Partition of SRAM is full and unable to immediately complete indirect operation

Quad SPI Flash Controller Programming Model

Setting Up the Quad SPI Flash Controller

The following steps describe how to set up the quad SPI controller:

1. Wait until any pending operation has completed.
2. Disable the quad SPI controller with the quad SPI enable field (`en`) of the `cfg` register.
3. Update the `instwidth` field of the `devrd` register with the instruction type you wish to use for indirect and direct writes and reads.
4. If mode bit enable bit (`enmodebits`) of the `devrd` register is enabled, update the mode bit register (`modebit`).
5. Update the `devsz` register as needed.
   Parts or all of this register might have been updated after initialization. The number of address bytes is a key configuration setting required for performing reads and writes. The number of bytes per page is required for performing any write. The number of bytes per device block is only required if the write protect feature is used.
6. Update the device delay register (`delay`).
   This register allows the user to adjust how the chip select is driven after each flash access. Each device may have different timing requirements. If the serial clock frequency is increased, these timing requirements become more critical. The numbers specified in this register are based on the period of the `qspi_ref_clk` clock. For example, some devices need 50 ns minimum time before the slave select can be reasserted after it has been deasserted. When the device is operating at 100 MHz, the clock period is 10 ns, so 40 ns extra is required. If the `qspi_ref_clk` clock is running at 400 MHz (2.5 ns period), specify a value of at least 16 to the clock delay for chip select deassert field (`nss`) of the `delay` register.
7. Update the `remapaddr` register as needed.
   This register only affects direct access mode.
8. Set up and enable the write protection registers (`wrprot`, `lowwrprot`, and `uppwrprot`) when write protection is required.
9. Enable required interrupts through the `irqmask` register.
10. Set up the `bauddiv` field of the `cfg` register to define the required clock frequency of the target device.
11. Update the read data capture register (`rddatacap`) if you need to change the auto-filled value.

    This register delays when the read data is captured and can help when the read data path from the device to the quad SPI controller is long and the device clock frequency is high.
12. Enable the quad SPI controller with the `en` field of the `cfg` register.

Related Information

Arria 10 Device Datasheet

Indirect Read Operation with DMA Disabled

The following steps describe the general software flow to set up the quad SPI controller for indirect read operation with the DMA disabled:

1. Perform the steps described in the **Setting Up the Quad SPI Flash Controller** on page 15-16 section.
2. Set the flash memory start address in the `indrdstaddr` register.
3. Set the number of bytes to be transferred in the `indrdcnt` register.
4. Set the indirect transfer trigger address in the `indaddrtrig` register.
5. Set up the required interrupts through the `irqmask` register.
6. If the watermark level is used, set the SRAM watermark level through the `indrdwater` register.
7. Start the indirect read operation by setting the `start` field of the `indrd` register to 1.
8. Either use the watermark level interrupt or poll the SRAM fill level in the `sramfill` register to determine when there is sufficient data in the SRAM.
9. Issue a read transaction to the indirect address to access the SRAM. Repeat **step 8** if more read transactions are needed to complete the indirect read transfer.
10. Either use the indirect complete interrupt to determine when the indirect read operation has completed or poll the completion status of the indirect read operation through the indirect completion status bit (`ind_ops_done_status`) of the `indrd` register.

Related Information

Setting Up the Quad SPI Flash Controller on page 15-16

Indirect Read Operation with DMA Enabled

The following steps describe the general software flow to set up the quad SPI controller for indirect read operation with the DMA enabled:

1. Perform the steps described in the **Setting Up the Quad SPI Flash Controller** on page 15-16 section.
2. Set the flash memory start address in the `indrdstaddr` register.
3. Set the number of bytes to be transferred in the `indrdcnt` register.
4. Set the indirect transfer trigger address in the `indaddrtrig` register.
5. Set the number of bytes for single and burst type DMA transfers in the `dmaper` register.
6. Optionally set the SRAM watermark level in the `indrdwater` register to control the rate DMA requests are issued.
7. Start an indirect read access by setting the `start` field of the `indrd` register to 1.
8. Either use the indirect complete interrupt to determine when the indirect read operation has completed or poll the completion status of the indirect read operation through the `ind_ops_done_status` field of the `indrd` register.

Related Information

Setting Up the Quad SPI Flash Controller on page 15-16

Indirect Write Operation with DMA Disabled

The following steps describe the general software flow to set up the quad SPI controller for indirect write operation with the DMA disabled:

1. Perform the steps described in the **Setting Up the Quad SPI Flash Controller** on page 15-16 section.
2. Set the flash memory start address in the `indwrstaddr` register.
3. Set up the number of bytes to be transferred in the `indwrcnt` register.
4. Set the indirect transfer trigger address in the `indaddrtrig` register.
5. Set up the required interrupts through the interrupt mask register (`irqmask`).
6. Optionally set the SRAM watermark level in the `indwrwater` register to control the rate DMA requests are issued. The value set must be greater than one flash page. For more information, refer to the **Indirect Write Operation** on page 15-7 section.
7. Start the indirect write operation by setting the `start` field of the `indwr` register to 1.
8. Either use the watermark level interrupt or poll the SRAM fill level in the `sramfill` register to determine when there is sufficient space in the SRAM.
9. Issue a write transaction to the indirect address to write one flash page of data to the SRAM. Repeat **step 8** if more write transactions are needed to complete the indirect write transfer. The final write may be less than one page of data.

Related Information

- **Indirect Write Operation** on page 15-7
- **Setting Up the Quad SPI Flash Controller** on page 15-16

Indirect Write Operation with DMA Enabled

The following steps describe the general software flow to set up the quad SPI controller for indirect write operation with the DMA enabled:

1. Perform the steps described in the **Setting Up the Quad SPI Flash Controller** on page 15-16 section.
2. Set the flash memory start address in the `indwrstaddr` register.
3. Set the number of bytes to be transferred in the `indcnt` field of the `indwr` register.
4. Set the indirect transfer trigger address in the `indaddrtrig` register.
5. Set the number of bytes for single and burst type DMA transfers in the `dmaper` register.
6. Optionally set the SRAM watermark level in the `indwrwater` register to control the rate DMA requests are issued. The value set must be greater than one flash page. For more information, refer to the **Indirect Write Operation** on page 15-7 section.
7. Start the indirect write access by setting the `start` field of the `indirwr` register to 1.
8. Either use the indirect complete interrupt to determine when the indirect write operation has completed or poll the completion status of the indirect write operation through the `ind_ops_done_status` field of the `indwr` register.

Related Information

- **Indirect Write Operation** on page 15-7
- **Setting Up the Quad SPI Flash Controller** on page 15-16

XIP Mode Operations

XIP mode is supported in most SPI flash devices. However, flash device manufacturers do not use a consistent standard approach. Most use signature bits that are sent to the device immediately following the address bytes. Some devices use signature bits and also require a flash device configuration register write to enable XIP mode.

Entering XIP Mode

Micron Quad SPI Flash Devices with Support for Basic-XIP

To enter XIP mode in a Micron quad SPI flash device with support for Basic-XIP, perform the following steps:

1. Save the values in the mode bits, if you intend to restore them upon exit.
2. Disable the direct access controller and indirect access controller to ensure no new read or write accesses are sent to the flash device.
3. Set the XIP mode bits in the `modebit` register to 0x80.
4. Enable the quad SPI controller's XIP mode by setting the `enterxipnextrd` bit of the `cfg` register to 1.
5. Re-enable the direct access controller and, if required, the indirect access controller.

Micron Quad SPI Flash Devices without Support for Basic-XIP

To enter XIP mode in a Micron quad SPI flash device without support for Basic-XIP, perform the following steps:

1. Save the values in the mode bits, if you intend to restore them upon exit.
2. Disable the direct access controller and indirect access controller to ensure no new read or write accesses will be sent to the flash device.
3. Ensure XIP mode is enabled in the flash device by setting the volatile configuration register (VCR) bit 3 to 1. Use the `flashcmd` register to issue the VCR write command.
4. Set the XIP mode bits in the `modebit` register to 0x00.
5. Enable the quad SPI controller's XIP mode by setting the `enterxipnextrd` bit of the `cfg` register to 1.
6. Re-enable the direct access controller and, if required, the indirect access controller.

Winbond Quad SPI Flash Devices

To enter XIP mode in a Winbond quad SPI flash device, perform the following steps:

1. Save the values in the mode bits, if you intend to restore them upon exit.
2. Disable the direct access controller and indirect access controller to ensure no new read or write accesses are sent to the flash device.
3. Set the XIP mode bits in the `modebit` register to 0x20.
4. Enable the quad SPI controller's XIP mode by setting the `enterxipnextrd` bit of the `cfg` register to 1.
5. Re-enable the direct access controller and, if required, the indirect access controller.

Spansion Quad SPI Flash Devices

To enter XIP mode a Spansion quad SPI flash device, perform the following steps:

1. Save the values in the mode bits, if you intend to restore them upon exit.
2. Disable the direct access controller and indirect access controller to ensure no new read or write accesses are sent to the flash device.
3. Set the XIP mode bits in the `modebit` register to 0xA0.
4. Enable the quad SPI controller's XIP mode by setting the `enterxipnextrd` bit of the `cfg` register to 1.
5. Re-enable the direct access controller and, if required, the indirect access controller.

Exiting XIP Mode

To exit XIP mode, perform the following steps:

1. Disable the direct access controller and indirect access controller to ensure no new read or write accesses are sent to the flash device.
2. Restore the mode bits to the values before entering XIP mode, depending on the flash device and manufacturer.
3. Set the `enterxipnextrd` bit of the `cfg` register to 0.

The flash device must receive a read instruction before it can disable its internal XIP mode state. Thus, XIP mode internally stays active until the next read instruction is serviced. Ensure that XIP mode is disabled before the end of any read sequence.

XIP Mode at Power on Reset

Some flash devices can be XIP-enabled as a nonvolatile configuration setting, allowing the flash device to enter XIP mode at power-on reset (POR) without software intervention. Software cannot discover the XIP state at POR through flash status register reads because an XIP-enabled flash device can only be accessed through the XIP read operation. If you known the device will enter XIP mode at POR, have your initial boot software configure the `modebit` register and set the `enterxipimm` bit of the `cfg` register to 1.

If you do not know in advance whether or not the device will enter XIP mode at POR, have your initial boot software issue an XIP mode exit command through the `flashcmd` register, then follow the steps in the *"Entering XIP Mode"* section. Software must be aware of the mode bit requirements of the device, because XIP mode entry and exit varies by device.

Related Information

Entering XIP Mode on page 15-19

Quad SPI Flash Controller Address Map and Register Definitions

For complete HPS address map and register definitions, refer to "Arria 10 HPS Address Map and Register Definitions".

Related Information

- **Error Checking and Correction Controller** on page 11-1
  For more information about how to program the ECC registers, refer to this chapter.
- **http://www.altera.com/literature/hb/arria-10/hps.html**

qspi_QSPIDATA Address Map

This address space is allocated for QSPI direct, indirect, and SPI legacy mode accesses. For more information, please refer to the Quad SPI Flash Controller chapter in the Arria 10 Hard Processor System Technical Reference Manual.

Module Instance	Base Address	End Address
i_qspi_QSPIDATA	0xFFA00000	0xFFAFFFFF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

qspiregs Address Map

Module Instance	Base Address	End Address
i_qspi_qspiregs	0xFF809000	0xFF8C07FF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
cfg on page 15-30	0x0	32	RW	0x80780000	QSPI Configuration Register
devrd on page 15-37	0x4	32	RW	0x3	Device Read Instruction Configuration Register
devwr on page 15-39	0x8	32	RW	0x2	Device Write Instruction Configuration Register
delay on page 15-41	0xC	32	RW	0x0	QSPI Device Delay Register
rddatacap on page 15-42	0x10	32	RW	0x1	Read Data Capture Register
devsz on page 15-43	0x14	32	RW	0x101002	Device Size Configuration Register

Register	Offset	Width	Access	Reset Value	Description
srampart on page 15-44	0x18	32	RW	0x80	SRAM Partition Configuration Register
indaddrtrig on page 15-45	0x1C	32	RW	0x0	Indirect AHB Address Trigger Register
dmaper on page 15-46	0x20	32	RW	0x0	DMA Peripheral Configuration Register
remapaddr on page 15-47	0x24	32	RW	0x0	Remap Address Register
modebit on page 15-48	0x28	32	RW	0x0	Mode Bit Configuration Register
sramfill on page 15-48	0x2C	32	RO	0x0	SRAM Fill Register
txthresh on page 15-49	0x30	32	RW	0x1	TX Threshold Register
rxthresh on page 15-50	0x34	32	RW	0x1	RX Threshold Register
irqstat on page 15-51	0x40	32	RW	0x100	Interrupt Status Register
irqmask on page 15-55	0x44	32	RW	0x0	Interrupt Mask
lowwrprot on page 15-57	0x50	32	RW	0x0	Lower Write Protection Register
uppwrprot on page 15-58	0x54	32	RW	0x0	Upper Write Protection Register
wrprot on page 15-59	0x58	32	RW	0x0	Write Protection Control Register
indrd on page 15-60	0x60	32	RW	0x0	Indirect Read Transfer Control Register
indrdwater on page 15-62	0x64	32	RW	0x0	Indirect Read Transfer Watermark Register
indrdstaddr on page 15-63	0x68	32	RW	0x0	Indirect Read Transfer Start Address Register
indrdcnt on page 15-64	0x6C	32	RW	0x0	Indirect Read Transfer Number Bytes Register

Register	Offset	Width	Access	Reset Value	Description
indwr on page 15-64	0x70	32	RW	0x0	Indirect Write Transfer Control Register
indwrwater on page 15-66	0x74	32	RW	0xFFFFFFFF	Indirect Write Transfer Watermark Register
indwrstaddr on page 15-67	0x78	32	RW	0x0	Indirect Write Transfer Start Address Register
indwrcnt on page 15-68	0x7C	32	RW	0x0	Indirect Write Transfer Number Bytes Register
flashcmd on page 15-69	0x90	32	RW	0x0	Flash Command Control Register
flashcmdaddr on page 15-73	0x94	32	RW	0x0	Flash Command Address Registers
flashcmdrddatalo on page 15-74	0xA0	32	RW	0x0	Flash Command Read Data Register (Lower)
flashcmdrddataup on page 15-75	0xA4	32	RW	0x0	Flash Command Read Data Register (Upper)
flashcmdwrdatalo on page 15-75	0xA8	32	RW	0x0	Flash Command Write Data Register (Lower)
flashcmdwrdataup on page 15-76	0xAC	32	RW	0x0	Flash Command Write Data Register (Upper)
moduleid on page 15-77	0xFC	32	RO	0x1001	Module ID Register

qspiregs Summary

Base Address: 0xFF809000

Register Address Offset	Bit Fields
i_qspi_qspiregs	

Register Address Offset	Bit Fields															
cfg 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	idle RO 0x1	config_resv2_fld RO 0x0								bauddiv RW 0xF				enterxipimm RW 0x0	enterxipnextrd RW 0x0	enahbremap RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	endma RW 0x0	wp RW 0x0	percslines RW 0x0				perseldec RW 0x0	enlegacyip RW 0x0	endiracc RW 0x0	config_resv1_fld RO 0x0				selclkphase RW 0x0	selclkpol RW 0x0	en RW 0x0
devrd 0x4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	rd_instr_resv5_fld RO 0x0			dummyrdclks RW 0x0					rd_instr_resv4_fld RO 0x0			enmodebits RW 0x0	rd_instr_resv3_fld RO 0x0		datawidth RW 0x0	
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rd_instr_resv2_fld RO 0x0		addrwidth RW 0x0		rd_instr_resv1_fld RO 0x0		instwidth RW 0x0		rdopcode RW 0x3							
devwr 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	wr_instr_resv4_fld RO 0x0			dummywrclks RW 0x0					wr_instr_resv3_fld RO 0x0						datawidth RW 0x0	
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	wr_instr_resv2_fld RO 0x0		addrwidth RW 0x0		wr_instr_resv1_fld RO 0x0				wropcode RW 0x2							
delay 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	nss RW 0x0								btwn RW 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	after RW 0x0								init RW 0x0							
rddatacap 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	rd_data_resv_fld RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rd_data_resv_fld RO 0x0											delay RW 0x0				byp RW 0x1

Register Address Offset	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
devsz 0x14	dev_size_resv_fld RO 0x0											bytespersubsector RW 0x10				
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	bytesperdevicepage RW 0x100												numaddrbytes RW 0x2			
srampart 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	resv_fld RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	resv_fld RO 0x0								addr RW 0x80							
indaddrtrig 0x1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addr RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addr RW 0x0															
dmaper 0x20	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	dma_periph_resv2_fld RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	dma_periph_resv2_fld RO 0x0				numburstreqbytes RW 0x0				dma_periph_resv1_fld RO 0x0				numsglreqbytes RW 0x0			
remapaddr 0x24	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	value RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	value RW 0x0															
modebit 0x28	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	mode_resv_fld RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	mode_resv_fld RO 0x0								mode RW 0x0							

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
sramfill 0x2C	indwrpart RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	indrdpart RO 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
txthresh 0x30	tx_thresh_resv_fld RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	tx_thresh_resv_fld RO 0x0												level RW 0x1			
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
rxthresh 0x34	rx_thresh_resv_fld RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rx_thresh_resv_fld RO 0x0												level RW 0x1			
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
irqstat 0x40	irq_stat_resv_fld RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	irq_stat_resv_fld RO 0x0			indsramfull RW 0x0	rxfull RW 0x0	rxthreshcmp RW 0x0	txfull RW 0x0	txthreshcmp RW 0x1	rxover RW 0x0	indxfrlvl RW 0x0	illegalacc RW 0x0	protwrattempt RW 0x0	indrdreject RW 0x0	indopdone RW 0x0	underflowdet RW 0x0	mode_m_fail_fld RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
irqmask 0x44	irq_mask_resv_fld RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	irq_mask_resv_fld RO 0x0			indsramfull RW 0x0	rxfull RW 0x0	rxthreshcmp RW 0x0	txfull RW 0x0	txthreshcmp RW 0x0	rxover RW 0x0	indxfrlvl RW 0x0	illegalacc RW 0x0	protwrattempt RW 0x0	indrdreject RW 0x0	indopdone RW 0x0	underflowdet RW 0x0	mode_m_fail_mask_fld RW 0x0

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	subsector RW 0x0															
lowwrprot 0x50	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	subsector RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	subsector RW 0x0															
uppwrprot 0x54	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	subsector RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	wr_prot_ctrl_resv_fld RO 0x0															
wrprot 0x58	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	wr_prot_ctrl_resv_fld RO 0x0														en RW 0x0	inv RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	indir_rd_xfer_resv_fld RO 0x0															
indrd 0x60	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	indir_rd_xfer_resv_fld RO 0x0								num_ind_ops_done RO 0x0		ind_ops_done_status RW 0x0	rd_queued RO 0x0	sram_full RW 0x0	rd_status RO 0x0	cancel WO 0x0	start WO 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	level RW 0x0															
indrdwater 0x64	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	level RW 0x0															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
indrdstaddr 0x68	addr RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addr RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
indrdcnt 0x6C	value RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	value RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
indwr 0x70	indir_wr_xfer_resv2_fld RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	indir_wr_xfer_resv2_fld RO 0x0								indcnt RO 0x0		indone RW 0x0	rdqueued RO 0x0	indir_wr_rsvd_fld RO 0x0	rdstat RO 0x0	cancel WO 0x0	start WO 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
indwrwater 0x74	level RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	level RW 0xFFFFFFFF															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
indwrstaddr 0x78	addr RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addr RW 0x0															

Register Address Offset	Bit Fields															
indwrcnt 0x7C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	value RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	value RW 0x0															
flashcmd 0x90	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cmdopcode RW 0x0								enrddata RW 0x0	numrddatabytes RW 0x0			encmdaddr RW 0x0	enmodebit RW 0x0	numaddrbytes RW 0x0	
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	enwrdata RW 0x0	numwrdatabytes RW 0x0			numdummybytes RW 0x0					flash_cmd_cntrl_resv1_fld RO 0x0					cmdexecstat RO 0x0	execcmd WO 0x0
flashcm-daddr 0x94	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addr RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addr RW 0x0															
flashcmdrd-datalo 0xA0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	data RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	data RW 0x0															
flashcmdrd-dataup 0xA4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	data RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	data RW 0x0															

Register Address Offset	Bit Fields															
flashcmdwr-datalo 0xA8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	data RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	data RW 0x0															
flashcmdwr-dataup 0xAC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	data RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	data RW 0x0															
moduleid 0xFC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	mod_id_resv_fld RO 0x0							value RO 0x1001								
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	value RO 0x1001															

cfg

Module Instance	Base Address	Register Address
i_qspi_qspiregs	0xFF809000	0xFF809000

Offset: 0x0

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
idle RO 0x1	config_resv2_fld RO 0x0								bauddiv RW 0xF				enterxipimm RW 0x0	enterxipnextrd RW 0x0	enahbremap RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
endma RW 0x0	wp RW 0x0	percslines RW 0x0				perseldec RW 0x0	enlegacyip RW 0x0	endiracc RW 0x0	config_resv1_fld RO 0x0				selclkphase RW 0x0	selclkpol RW 0x0	en RW 0x0

cfg Fields

Bit	Name	Description	Access	Reset
31	idle	This is a STATUS read-only bit. Note this is a retimed signal, so there will be some inherent delay on the generation of this status signal. **Value** **Description** 0 NOTSET 1 SET	RO	0x1
30:23	config_resv2_fld		RO	0x0

Bit	Name	Description	Access	Reset
22:19	bauddiv	SPI baud rae = (master reference clock) baud_rate_divisor Value — Description 0 — BAUD2 1 — BAUD4 2 — BAUD6 3 — BAUD8 4 — BAUD10 5 — BAUD12 6 — BAUD14 7 — BAUD16 8 — BAUD18 9 — BAUD20 10 — BAUD22 11 — BAUD24 12 — BAUD26 13 — BAUD28 14 — BAUD30 15 — BAUD32	RW	0xF

Bit	Name	Description	Access	Reset
18	enterxipimm	Value=0 : If XIP is enabled, then setting to 0 will cause the controller to exit XIP mode on the next READ instruction. Value=1 : Operate the device in XIP mode immediately Use this register when the external device wakes up in XIP mode (as per the contents of its non- volatile configuration register). The controller will assume the next READ instruction will be passed to the device as an XIP instruction, and therefore will not require the READ opcode to be transferred. Note: To exit XIP mode, this bit should be set to 0. This will take effect in the attached device only after the next READ instruction is executed. Software therefore should ensure that at least one READ instruction is requested after resetting this bit in order to be sure that XIP mode is exited. **Value** **Description** 0 DISABLE 1 ENABLE	RW	0x0

Bit	Name	Description	Access	Reset
17	enterxipnextrd	Value=0 : If XIP is enabled, then setting to 0 will cause the controller to exit XIP mode on the next READ instruction. Value=1 : If XIP is disabled, then setting to ? 1? will inform the controller that the device is ready to enter XIP on the next READ instruction. The controller will therefore send the appropriate command sequence, including mode bits to cause the device to enter XIP mode. Use this register after the controller has ensured the FLASH device has been configured to be ready to enter XIP mode. Note : To exit XIP mode, this bit should be set to 0. This will take effect in the attached device only AFTER the next READ instruction is executed. Software should therefore ensure that at least one READ instruction is requested after resetting this bit before it can be sure XIP mode in the device is exited. **Value** **Description** 0 DISABLE 1 ENABLE	RW	0x0
16	enahbremap	(Direct Access Mode Only) When set to 1, the incoming AHB address will be adapted and sent to the FLASH device as (address + N), where N is the value stored in the remap address register. **Value** **Description** 0 DISABLE 1 ENABLE	RW	0x0

Bit	Name	Description	Access	Reset
15	endma	Set to 1 to enable the DMA handshaking logic. When enabled the QSPI will trigger DMA transfer requests via the DMA peripheral interface. Set to 0 to disable **Value** / **Description** 0 DISABLE 1 ENABLE	RW	0x0
14	wp	Set to drive the Write Protect pin of the FLASH device. This is resynchronized to the generated memory clock as necessary. **Value** / **Description** 0 WRTPROTOFF 1 WRPROTON	RW	0x0
13:10	percslines	Peripheral chip select lines If pdec = 0, ss[3:0] are output thus: ss[3:0] n_ss_out[3:0] xxx0 1110 xx01 1101 x011 1011 0111 0111 1111 1111 (no peripheral selected) else ss[3:0] directly drives n_ss_out[3:0]	RW	0x0
9	perseldec	0 : only 1 of 4 selects n_ss_out[3:0] is active 1 : allow external 4-to-16 decode (n_ss_out = ss) **Value** / **Description** 0 SEL1OF4 1 SEL4TO16	RW	0x0

Bit	Name	Description	Access	Reset
8	enlegacyip	0 : Use Direct Access Controller/ Indirect Access Controller 1 : legacy Mode is enabled. In this mode, any write to the controller via the AHB interface is serialized and sent to the FLASH device. Any valid AHB read will pop the internal RX-FIFO, retrieving data that was forwarded by the external FLASH device on the SPI lines,4,2 or 1 byte transfers are permitted and controlled via the HSIZE input. **Value** **Description** 0 DIMODE 1 LEGMODE	RW	0x0
7	endiracc	0 : disable the Direct Access Controller once current transfer of the data word (FF_W) is complete. 1 : enable the Direct Access Controller When the Direct Access Controller and Indirect Access Controller are both disabled, all AHB requested are completed with an error response. **Value** **Description** 0 DISABLE 1 ENABLE	RW	0x0
6:3	config_resv1_fld		RO	0x0
2	selclkphase	Selects whether the clock is in an active or inactive phase outside the SPI word. 0 : the SPI clock is active outside the word 1 : the SPI clock is inactive outside the word **Value** **Description** 0 ACTIVE 1 INACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
1	selclkpol	0 : the SPI clock is quiescent low 1 : the SPI clock is quiescent high **Value** **Description** 0 HIGH 1 LOW	RW	0x0
0	en	0 : disable the QSPI once current transfer of the data word (FF_W) is complete. 1 : enable the QSPI When spi_enable = 0, all output enables are inactive and all pins are set to input mode. **Value** **Description** 0 DISABLE 1 ENABLE	RW	0x0

devrd

Module Instance	Base Address	Register Address
i_qspi_qspiregs	0xFF809000	0xFF809004

Offset: 0x4

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
rd_instr_resv5_fld RO 0x0			dummyrdclks RW 0x0					rd_instr_resv4_fld RO 0x0			enmodebits RW 0x0	rd_instr_resv3_fld RO 0x0		datawidth RW 0x0	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rd_instr_resv2_fld RO 0x0		addrwidth RW 0x0		rd_instr_resv1_fld RO 0x0		instwidth RW 0x0		rdopcode RW 0x3							

devrd Fields

Bit	Name	Description	Access	Reset
31:29	rd_instr_resv5_fld		RO	0x0
28:24	dummyrdclks	Number of dummy clock cycles required by device for read instruction.	RW	0x0
23:21	rd_instr_resv4_fld		RO	0x0
20	enmodebits	Set this field to 1 to ensure that the mode bits as defined in the Mode Bit Configuration register are sent following the address bytes. **Value** — **Description** 0 — NOORDER 1 — ORDER	RW	0x0
19:18	rd_instr_resv3_fld		RO	0x0
17:16	datawidth	0 : SIO mode data is shifted to the device on DQ0 only and from the device on DQ1 only 1 : Used for Dual Input/Output instructions. For data transfers, DQ0 and DQ1 are used as both inputs and outputs. 2 : Used for Quad Input/Output instructions. For data transfers, DQ0,DQ1,DQ2 and DQ3 are used as both inputs and outputs. **Value** — **Description** 0 — SINGLE 1 — DUAL 2 — QUAD	RW	0x0
15:14	rd_instr_resv2_fld		RO	0x0

Bit	Name	Description	Access	Reset
13:12	addrwidth	0 : Addresses can be shifted to the device on DQ0 only 1 : Addresses can be shifted to the device on DQ0 and DQ1 only 2 : Addresses can be shifted to the device on DQ0, DQ1, DQ2 and DQ3 **Value** **Description** 0 SINGLE 1 DUAL 2 QUAD	RW	0x0
11:10	rd_instr_resv1_fld		RO	0x0
9:8	instwidth	0 : Use Standard SPI mode (instruction always shifted into the device on DQ0 only) 1 : Use DIO-SPI mode (Instructions, Address and Data always sent on DQ0 and DQ1) 2 : Use QIO-SPI mode (Instructions, Address and Data always sent on DQ0, DQ1, DQ2 and DDQ3) **Value** **Description** 0 SINGLE 1 DUAL 2 QUAD	RW	0x0
7:0	rdopcode	Read Opcode to use when not in XIP mode **Value** **Description** 3 READ 11 FASTREAD	RW	0x3

devwr

Module Instance	Base Address	Register Address
i_qspi_qspiregs	0xFF809000	0xFF809008

Offset: 0x8

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
wr_instr_resv4_fld RO 0x0			dummywrclks RW 0x0					wr_instr_resv3_fld RO 0x0						datawidth RW 0x0	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
wr_instr_resv2_fld RO 0x0		addrwidth RW 0x0		wr_instr_resv1_fld RO 0x0				wropcode RW 0x2							

devwr Fields

Bit	Name	Description	Access	Reset
31:29	wr_instr_resv4_fld		RO	0x0
28:24	dummywrclks	Number of dummy clock cycles required by device for write instruction.	RW	0x0
23:18	wr_instr_resv3_fld		RO	0x0
17:16	datawidth	0 : SIO mode data is shifted to the device on DQ0 only and from the device on DQ1 only 1 : Used for Dual Input/Output instructions. For data transfers, DQ0 and DQ1 are used as both inputs and outputs. 2 : Used for Quad Input/Output instructions. For data transfers, DQ0,DQ1,DQ2 and DQ3 are used as both inputs and outputs. **Value** — **Description** 0 — SINGLE 1 — DUAL 2 — QUAD	RW	0x0
15:14	wr_instr_resv2_fld		RO	0x0

Bit	Name	Description	Access	Reset
13:12	addrwidth	0 : Addresses can be shifted to the device on DQ0 only 1 : Addresses can be shifted to the device on DQ0 and DQ1 only 2 : Addresses can be shifted to the device on DQ0, DQ1, DQ2 and DQ3 Value: Description 0: SINGLE 1: DUAL 2: QUAD	RW	0x0
11:8	wr_instr_resv1_fld		RO	0x0
7:0	wropcode	Write Opcode	RW	0x2

delay

This register is used to introduce relative delays into the generation of the master output signals. All timings are defined in cycles of the SPI REFERENCE CLOCK/ext_clk, defined in this table as SPI master ref clock.

Module Instance	Base Address	Register Address
i_qspi_qspiregs	0xFF809000	0xFF80900C

Offset: 0xC

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
nss RW 0x0								btwn RW 0x0							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
after RW 0x0								init RW 0x0							

delay Fields

Bit	Name	Description	Access	Reset
31:24	nss	Delay in master reference clocks for the length that the master mode chip select outputs are de-asserted between transactions. The minimum delay is always SCLK period to ensure the chip select is never re-asserted within an SCLK period.	RW	0x0
23:16	btwn	Delay in master reference clocks between one chip select being de-activated and the activation of another. This is used to ensure a quiet period between the selection of two different slaves and requires the transmit FIFO to be empty.	RW	0x0
15:8	after	Delay in QSPI_CLK clocks between the last bit of the current transaction and deasserting the device chip select (QSPI_SS). By default (when this field is 0x0), QSPI_SS is deasserted on the last falling edge of SCLK_OUT at the completion of the current transaction. When this field is programmed with N, QSPI_SS is deasserted N QSPI_CLK clocks after the last falling edge of SCLK_OUT.	RW	0x0
7:0	init	Delay in master reference clocks between setting n_ss_out low and first bit transfer.	RW	0x0

rddatacap

Module Instance	Base Address	Register Address
i_qspi_qspiregs	0xFF809000	0xFF809010

Offset: 0x10

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
rd_data_resv_fld RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rd_data_resv_fld RO 0x0											delay RW 0x0				byp RW 0x1

rddatacap Fields

Bit	Name	Description	Access	Reset
31:5	rd_data_resv_fld		RO	0x0
4:1	delay	Delay the read data capturing logic by the programmed number of ref_clk cycles	RW	0x0
0	byp	Bypass the adapted loopback clock circuit Value: Description 0: BYPASS 1: NOBYPASS	RW	0x1

devsz

Module Instance	Base Address	Register Address
i_qspi_qspiregs	0xFF809000	0xFF809014

Offset: 0x14

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dev_size_resv_fld RO 0x0											bytespersubsector RW 0x10				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
bytesperdevicepage RW 0x100												numaddrbytes RW 0x2			

devsz Fields

Bit	Name	Description	Access	Reset
31:21	dev_size_resv_fld		RO	0x0
20:16	bytespersubsector	Number of bytes per Block. This is required by the controller for performing the write protection logic. The number of bytes per block must be a power of 2 number.	RW	0x10
15:4	bytesperdevicepage	Number of bytes per device page. This is required by the controller for performing FLASH writes up to and across page boundaries.	RW	0x100
3:0	numaddrbytes	Number of address bytes. A value of 0 indicates 1 byte.	RW	0x2

srampart

Module Instance	Base Address	Register Address
i_qspi_qspiregs	0xFF809000	0xFF809018

Offset: 0x18

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
resv_fld RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
resv_fld RO 0x0								addr RW 0x80							

srampart Fields

Bit	Name	Description	Access	Reset
31:8	resv_fld		RO	0x0
7:0	addr	Defines the size of the indirect read partition in the SRAM, in units of SRAM locations. By default, half of the SRAM is reserved for indirect read operation, and half for indirect write. The size of this register will scale with the depth of the SRAM.	RW	0x80

indaddrtrig

Module Instance	Base Address	Register Address
i_qspi_qspiregs	0xFF809000	0xFF80901C

Offset: 0x1C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addr RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addr RW 0x0															

indaddrtrig Fields

Bit	Name	Description	Access	Reset
31:0	addr	This is the base address that will be used by the AHB controller. When the incoming AHB read access address matches a range of addresses from this trigger address to the trigger address + 15, then the AHB request will be completed by fetching data from the Indirect Controllers SRAM.	RW	0x0

dmaper

Module Instance	Base Address	Register Address
i_qspi_qspiregs	0xFF809000	0xFF809020

Offset: 0x20

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dma_periph_resv2_fld RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
dma_periph_resv2_fld RO 0x0				numburstreqbytes RW 0x0				dma_periph_resv1_fld RO 0x0				numsglreqbytes RW 0x0			

dmaper Fields

Bit	Name	Description	Access	Reset
31:12	dma_periph_resv2_fld		RO	0x0

Bit	Name	Description	Access	Reset
11:8	numburstreqbytes	Number of bytes in a burst type request on the DMA peripheral request. A programmed value of 0 represents a single byte. This should be setup before starting the indirect read or write operation. The actual number of bytes used is 2**(value in this register) which will simplify implementation.	RW	0x0
7:4	dma_periph_resv1_fld		RO	0x0
3:0	numsglreqbytes	Number of bytes in a single type request on the DMA peripheral request. A programmed value of 0 represents a single byte. This should be setup before starting the indirect read or write operation. The actual number of bytes used is 2**(value in this register) which will simplify implementation.	RW	0x0

remapaddr

Module Instance	Base Address	Register Address
i_qspi_qspiregs	0xFF809000	0xFF809024

Offset: 0x24

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
value RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
value RW 0x0															

remapaddr Fields

Bit	Name	Description	Access	Reset
31:0	value	This register is used to remap an incoming AHB address to a different address used by the FLASH device.	RW	0x0

modebit

Module Instance	Base Address	Register Address
i_qspi_qspiregs	0xFF809000	0xFF809028

Offset: 0x28

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
mode_resv_fld RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
mode_resv_fld RO 0x0								mode RW 0x0							

modebit Fields

Bit	Name	Description	Access	Reset
31:8	mode_resv_fld		RO	0x0
7:0	mode	These are the 8 mode bits that are sent to the device following the address bytes if mode bit transmission has been enabled.	RW	0x0

sramfill

Module Instance	Base Address	Register Address
i_qspi_qspiregs	0xFF809000	0xFF80902C

Offset: 0x2C

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
indwrpart RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
indrdpart RO 0x0															

sramfill Fields

Bit	Name	Description	Access	Reset
31:16	indwrpart	Identifies the current fill level of the SRAM Indirect Write partition	RO	0x0
15:0	indrdpart	Identifies the current fill level of the SRAM Indirect Read partition	RO	0x0

txthresh

Module Instance	Base Address	Register Address
i_qspi_qspiregs	0xFF809000	0xFF809030

Offset: 0x30

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
tx_thresh_resv_fld RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
tx_thresh_resv_fld RO 0x0												level RW 0x1			

txthresh Fields

Bit	Name	Description	Access	Reset
31:4	tx_thresh_resv_fld		RO	0x0
3:0	level	Defines the level at which the small TX FIFO not full interrupt is generated	RW	0x1

rxthresh

Device Instruction Configuration Register

Module Instance	Base Address	Register Address
i_qspi_qspiregs	0xFF809000	0xFF809034

Offset: 0x34

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
rx_thresh_resv_fld RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rx_thresh_resv_fld RO 0x0												level RW 0x1			

rxthresh Fields

Bit	Name	Description	Access	Reset
31:4	rx_thresh_resv_fld		RO	0x0

Bit	Name	Description	Access	Reset
3:0	level	Defines the level at which the small RX FIFO not empty interrupt is generated	RW	0x1

irqstat

The status fields in this register are set when the described event occurs and the interrupt is enabled in the mask register. When any of these bit fields are set, the interrupt output is asserted high. The fields are each cleared by writing a 1 to the field. Note that bit fields 6 thru 10 are only valid when legacy SPI mode is active.

Module Instance	Base Address	Register Address
i_qspi_qspiregs	0xFF809000	0xFF809040

Offset: 0x40

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
irq_stat_resv_fld RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
irq_stat_resv_fld RO 0x0			indsramfull RW 0x0	rxfull RW 0x0	rxthreshcmp RW 0x0	txfull RW 0x0	txthreshcmp RW 0x1	rxovr RW 0x0	indxfrlvl RW 0x0	illegalacc RW 0x0	protwrattempt RW 0x0	indrdreject RW 0x0	indopdone RW 0x0	underflowdet RW 0x0	mode_m_fail_fld RW 0x0

irqstat Fields

Bit	Name	Description	Access	Reset
31:13	irq_stat_resv_fld		RO	0x0

Bit	Name	Description	Access	Reset
12	indsramfull	Indirect Read Partition of SRAM is full and unable to immediately complete indirect operation Value: 0 — Description: RDPARTNOTFULL Value: 1 — Description: RDPARTFULL	RW	0x0
11	rxfull	Current FIFO status can be ignored in non-SPI legacy mode 0 : FIFO is not full 1 : FIFO is full Value: 0 — Description: NOTFULL Value: 1 — Description: FULL	RW	0x0
10	rxthreshcmp	Current FIFO status can be ignored in non-SPI legacy mode 0 : FIFO has less than RX THRESHOLD entries, 1 : FIFO has >= THRESHOLD entries Value: 0 — Description: LE Value: 1 — Description: GT	RW	0x0
9	txfull	Current FIFO status can be ignored in non-SPI legacy mode 0 : FIFO is not full, 1 : FIFO is full Value: 0 — Description: NOTFULL Value: 1 — Description: FULL	RW	0x0
8	txthreshcmp	Current FIFO status can be ignored in non-SPI legacy mode 0 : FIFO has >= THRESHOLD entries, 1 : FIFO has less than THRESHOLD entries Value: 0 — Description: GT Value: 1 — Description: LE	RW	0x1

Bit	Name	Description	Access	Reset
7	rxover	This should only occur in Legacy SPI mode. Set if an attempt is made to push the RX FIFO when it is full. This bit is reset only by a system reset and cleared only when this register is read. If a new push to the RX FIFO occurs coincident with a register read this flag will remain set. 0 : no overflow has been detected. 1 : an overflow has occurred. **Value** **Description** 0 NORCVOVER 1 RCVOVER	RW	0x0
6	indxfrlvl	Indirect Transfer Watermark Level Breached **Value** **Description** 0 NOWATERLVL 1 WATERLEVL	RW	0x0
5	illegalacc	Illegal AHB access has been detected. AHB wrapping bursts and the use of SPLIT/RETRY accesses will cause this error interrupt to trigger. **Value** **Description** 0 NOILLEGALAHB 1 ILLEGALAHB	RW	0x0
4	protwrattempt	Write to protected area was attempted and rejected. **Value** **Description** 0 NOWRITEPROT 1 WRITEPROT	RW	0x0

Bit	Name	Description	Access	Reset
3	indrdreject	Indirect operation was requested but could not be accepted. Two indirect operations already in storage. **Value** **Description** 0 NOINDIRECTREQ 1 INDIRECTREQ	RW	0x0
2	indopdone	Controller has completed last triggered indirect operation **Value** **Description** 0 NOINDIRECTOP 1 INDIRECTOP	RW	0x0
1	underflowdet	0 : no underflow has been detected 1 : underflow is detected and an attempt to transfer data is made when the small TX FIFO is empty. This may occur when AHB write data is being supplied too slowly to keep up with the requested write operation This bit is reset only by a system reset and cleared only when the register is read. **Value** **Description** 0 NOUNDERFLOW 1 UNDERFLOW	RW	0x0
0	mode_m_fail_fld	Mode M failure indicates the voltage on pin n_ss_in is inconsis-tent with the SPI mode. Set =1 if n_ss_in is low in master mode (multi-master contention). These conditions will clear the spi_enable bit and disable the SPI. This bit is reset only by a system reset and cleared only when this register is read. 0 : no mode fault has been detected 1 : a mode fault has occurred	RW	0x0

irqmask

0 : the interrupt for the corresponding interrupt status register bit is disabled. 1 : the interrupt for the corresponding interrupt status register bit is enabled.

Module Instance	Base Address	Register Address
i_qspi_qspiregs	0xFF809000	0xFF809044

Offset: 0x44

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
irq_mask_resv_fld RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
irq_mask_resv_fld RO 0x0			indsramfull RW 0x0	rxfull RW 0x0	rxthreshcmp RW 0x0	txfull RW 0x0	txthreshcmp RW 0x0	rxoverr RW 0x0	indxfrlvl RW 0x0	illegalacc RW 0x0	protwrattempt RW 0x0	indrdreject RW 0x0	indopdone RW 0x0	underflowdet RW 0x0	mode_m_fail_mask_fld RW 0x0

irqmask Fields

Bit	Name	Description	Access	Reset
31:13	irq_mask_resv_fld		RO	0x0
12	indsramfull	Value: Description 0: DISABLED 1: ENABLED	RW	0x0
11	rxfull	Value: Description 0: DISABLED 1: ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
10	rxthreshcmp	**Value** **Description** 0 DISABLED 1 ENABLED	RW	0x0
9	txfull	**Value** **Description** 0 DISABLED 1 ENABLED	RW	0x0
8	txthreshcmp	**Value** **Description** 0 DISABLED 1 ENABLED	RW	0x0
7	rxover	**Value** **Description** 0 DISABLED 1 ENABLED	RW	0x0
6	indxfrlvl	**Value** **Description** 0 DISABLED 1 ENABLED	RW	0x0
5	illegalacc	**Value** **Description** 0 DISABLED 1 ENABLED	RW	0x0
4	protwrattempt	**Value** **Description** 0 DISABLED 1 ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
3	indrdreject	Value: 0 DISABLED; 1 ENABLED	RW	0x0
2	indopdone	Value: 0 DISABLED; 1 ENABLED	RW	0x0
1	underflowdet	Value: 0 DISABLED; 1 ENABLED	RW	0x0
0	mode_m_fail_mask_fld		RW	0x0

lowwrprot

Module Instance	Base Address	Register Address
i_qspi_qspiregs	0xFF809000	0xFF809050

Offset: 0x50

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
subsector RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
subsector RW 0x0															

lowwrprot Fields

Bit	Name	Description	Access	Reset
31:0	subsector	The block number that defines the lower block in the range of blocks that is to be locked from writing. The definition of a block in terms of number of bytes is programmable via the Device Size Configuration register.	RW	0x0

uppwrprot

Module Instance	Base Address	Register Address
i_qspi_qspiregs	0xFF809000	0xFF809054

Offset: 0x54

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
subsector RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
subsector RW 0x0															

uppwrprot Fields

Bit	Name	Description	Access	Reset
31:0	subsector	The block number that defines the upper block in the range of blocks that is to be locked from writing. The definition of a block in terms of number of bytes is programmable via the Device Size Configuration register.	RW	0x0

wrprot

Module Instance	Base Address	Register Address
i_qspi_qspiregs	0xFF809000	0xFF809058

Offset: 0x58

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
wr_prot_ctrl_resv_fld RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
wr_prot_ctrl_resv_fld RO 0x0														en RW 0x0	inv RW 0x0

wrprot Fields

Bit	Name	Description	Access	Reset
31:2	wr_prot_ctrl_resv_fld		RO	0x0
1	en	When set to 1, any AHB write access with an address within the protection region defined in the lower and upper write protection registers is rejected. An AHB error response is generated and an interrupt source triggered. When set to 0, the protection region is disabled. **Value** **Description** 0 DISABLE 1 ENABLE	RW	0x0

Bit	Name	Description	Access	Reset
0	inv	When set to 1, the protection region defined in the lower and upper write protection registers is inverted meaning it is the region that the system is permitted to write to. When set to 0, the protection region defined in the lower and upper write protection registers is the region that the system is not permitted to write to. **Value** **Description** 0 DISABLE 1 ENABLE	RW	0x0

indrd

Module Instance	Base Address	Register Address
i_qspi_qspiregs	0xFF809000	0xFF809060

Offset: 0x60

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
indir_rd_xfer_resv_fld RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
indir_rd_xfer_resv_fld RO 0x0								num_ind_ops_done RO 0x0		ind_ops_done_status RW 0x0	rd_queued RO 0x0	sram_full RW 0x0	rd_status RO 0x0	cancel WO 0x0	start WO 0x0

indrd Fields

Bit	Name	Description	Access	Reset
31:8	indir_rd_xfer_resv_fld		RO	0x0
7:6	num_ind_ops_done	This field contains the number of indirect operations which have been completed. This is used in conjunction with the indirect completion status field (bit 5). It is incremented by hardware when an indirect operation has completed. Write a 1 to bit 5 of this register to decrement it.	RO	0x0
5	ind_ops_done_status	This field is set to 1 when an indirect operation has completed. Write a 1 to this field to clear it. **Value** — **Description** 0 — NOACTION 1 — INDCOMP	RW	0x0
4	rd_queued	Two indirect read operations have been queued **Value** — **Description** 0 — NOACTION 1 — QUINDIRECTRD	RO	0x0
3	sram_full	SRAM full and unable to immediately complete an indirect operation. Write a 1 to this field to clear it."; indirect operation (status) **Value** — **Description** 0 — NOACTION 1 — SRAMFULL	RW	0x0

Bit	Name	Description	Access	Reset
2	rd_status	Indirect read operation in progress (status) Value: Description 0: NOACTION 1: READOP	RO	0x0
1	cancel	Writing a 1 to this bit will cancel all ongoing indirect read operations. Value: Description 0: NOACTION 1: CANCEL	WO	0x0
0	start	Writing a 1 to this bit will trigger an indirect read operation. The assumption is that the indirect start address and the indirect number of bytes register is setup before triggering the indirect read operation. Value: Description 0: DISABLED 1: ENABLED	WO	0x0

indrdwater

Module Instance	Base Address	Register Address
i_qspi_qspiregs	0xFF809000	0xFF809064

Offset: 0x64

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
level RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
level RW 0x0															

indrdwater Fields

Bit	Name	Description	Access	Reset
31:0	level	This represents the minimum fill level of the SRAM before a DMA peripheral access is permitted. When the SRAM fill level passes the watermark, an interrupt is also generated. This field can be disabled by writing a value of all zeroes.	RW	0x0

indrdstaddr

Module Instance	Base Address	Register Address
i_qspi_qspiregs	0xFF809000	0xFF809068

Offset: 0x68

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addr RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addr RW 0x0															

indrdstaddr Fields

Bit	Name	Description	Access	Reset
31:0	addr	This is the start address from which the indirect access will commence its READ operation.	RW	0x0

indrdcnt

Module Instance	Base Address	Register Address
i_qspi_qspiregs	0xFF809000	0xFF80906C

Offset: 0x6C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
value RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
value RW 0x0															

indrdcnt Fields

Bit	Name	Description	Access	Reset
31:0	value	This is the number of bytes that the indirect access will consume. This can be bigger than the configured size of SRAM.	RW	0x0

indwr

Module Instance	Base Address	Register Address
i_qspi_qspiregs	0xFF809000	0xFF809070

Offset: 0x70

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
indir_wr_xfer_resv2_fld RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
indir_wr_xfer_resv2_fld RO 0x0								indcnt RO 0x0		inddone RW 0x0	rdqueued RO 0x0	indir_wr_rsvd_fld RO 0x0	rdstat RO 0x0	cancel WO 0x0	start WO 0x0

indwr Fields

Bit	Name	Description	Access	Reset
31:8	indir_wr_xfer_resv2_fld		RO	0x0
7:6	indcnt	This field contains the number of indirect operations which have been completed. This is used in conjunction with the indirect completion status field (bit 5). It is incremented by hardware when an indirect operation has completed. Write a 1 to bit 5 of this register to decrement it.	RO	0x0
5	inddone	This field is set to 1 when an indirect operation has completed. Write a 1 to this field to clear it. Value — Description 0 — NOACTION 1 — INDCOMPST	RW	0x0

Bit	Name	Description	Access	Reset
4	rdqueued	Two indirect write operations have been queued Value: 0 — NOACTION Value: 1 — INDWROP	RO	0x0
3	indir_wr_rsvd_fld		RO	0x0
2	rdstat	Indirect write operation in progress (status) Value: 0 — NOACTION Value: 1 — INDWRSTAT	RO	0x0
1	cancel	Writing a 1 to this bit will cancel all ongoing indirect write operations. Value: 0 — NOACTION Value: 1 — CANCEINDWR	WO	0x0
0	start	Writing a 1 to this bit will trigger an indirect write operation. The assumption is that the indirect start address and the indirect number of bytes register is setup before triggering the indirect write operation. Value: 0 — DISABLED Value: 1 — ENABLED	WO	0x0

indwrwater

Module Instance	Base Address	Register Address
i_qspi_qspiregs	0xFF809000	0xFF809074

Offset: 0x74

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
level RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
level RW 0xFFFFFFFF															

indwrwater Fields

Bit	Name	Description	Access	Reset
31:0	level	This represents the maximum fill level of the SRAM before a DMA peripheral access is permitted. When the SRAM fill level falls below the watermark, an interrupt is also generated. This field can be disabled by writing a value of all ones.	RW	0xFFFFFFFF

indwrstaddr

Module Instance	Base Address	Register Address
i_qspi_qspiregs	0xFF809000	0xFF809078

Offset: 0x78

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addr RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addr RW 0x0															

indwrstaddr Fields

Bit	Name	Description	Access	Reset
31:0	addr	This is the start address from which the indirect access will commence its READ operation.	RW	0x0

indwrcnt

Module Instance	Base Address	Register Address
i_qspi_qspiregs	0xFF809000	0xFF80907C

Offset: 0x7C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
value RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
value RW 0x0															

indwrcnt Fields

Bit	Name	Description	Access	Reset
31:0	value	This is the number of bytes that the indirect access will consume. This can be bigger than the configured size of SRAM.	RW	0x0

flashcmd

Module Instance	Base Address	Register Address
i_qspi_qspiregs	0xFF809000	0xFF809090

Offset: 0x90

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cmdopcode RW 0x0								enrddata RW 0x0	numrddatabytes RW 0x0			encmdaddr RW 0x0	enmodebit RW 0x0	numaddrbytes RW 0x0	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
enwrdata RW 0x0	numwrdatabytes RW 0x0			numdummybytes RW 0x0				flash_cmd_cntrl_resv1_fld RO 0x0						cmdexecstat RO 0x0	execcmd WO 0x0

flashcmd Fields

Bit	Name	Description	Access	Reset
31:24	cmdopcode	The command opcode field should be setup before triggering the command. For example, 0x20 maps to SubSector Erase. Writeing to the execute field (bit 0) of this register launches the command. NOTE : Using this approach to issue commands to the device will make use of the instruction type of the device instruction configuration register. If this field is set to 2'b00, then the command opcode, command address, command dummy bytes and command data will all be transferred in a serial fashion. If this field is set to 2'b01, then the command opcode, command address, command dummy bytes and command data will all be transferred in parallel using DQ0 and DQ1 pins. If this field is set to 2'b10, then the command opcode, command address, command dummy bytes and command data will all be transferred in parallel using DQ0, DQ1, DQ2 and DQ3 pins.	RW	0x0
23	enrddata	Set to 1 if the command specified in the command opcode field (bits 31:24) requires read data bytes to be received from the device. Value: 0 — Description: NOACTION Value: 1 — Description: ENABLE	RW	0x0

Bit	Name	Description	Access	Reset
22:20	numrddatabytes	Up to 8 data bytes may be read using this command. Set to 0 for 1 byte and 7 for 8 bytes. **Value** — **Description** 0 — RDBYTE1 1 — RDBYTE2 2 — RDBYTE3 3 — RDBYTE4 4 — RDBYTE5 5 — RDBYTE6 6 — RDBYTE7 7 — RDBYTE8	RW	0x0
19	encmdaddr	Set to 1 if the command specified in bits 31:24 requires an address. This should be setup before triggering the command via writing a 1 to the execute field. **Value** — **Description** 0 — DISABLED 1 — ENABLED	RW	0x0
18	enmodebit	Set to 1 to ensure the mode bits as defined in the Mode Bit Configuration register are sent following the address bytes. **Value** — **Description** 0 — DISABLED 1 — ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
17:16	numaddrbytes	Set to the number of address bytes required (the address itself is programmed in the FLASH COMMAND ADDRESS REGISTERS). This should be setup before triggering the command via bit 0 of this register. 2'b00 : 1 address byte 2'b01 : 2 address bytes 2'b10 : 3 address bytes 2'b11 : 4 address bytes **Value** — **Description** 0 — ADDRBYTE1 1 — ADDRBYTE2 2 — ADDRBYTE3 3 — ADDRBYTE4	RW	0x0
15	enwrdata	Set to 1 if the command specified in the command opcode field requires write data bytes to be sent to the device. **Value** — **Description** 0 — NOACTION 1 — WRDATABYTES	RW	0x0
14:12	numwrdatabytes	Up to 8 Data bytes may be written using this command Set to 0 for 1 byte, 7 for 8 bytes. **Value** — **Description** 0 — WRBYTE1 1 — WRBYTE2 2 — WRBYTE3 3 — WRBYTE4 4 — WRBYTE5 5 — WRBYTE6 6 — WRBYTE7 7 — WRBYTE8	RW	0x0
11:7	numdummybytes	Set to the number of dummy bytes required This should be setup before triggering the command via the execute field of this register.	RW	0x0

Bit	Name	Description	Access	Reset
6:2	flash_cmd_cntrl_resv1_fld		RO	0x0
1	cmdexecstat	Command execution in progress. Value: Description 0: NOACTION 1: EXECUTESTAT	RO	0x0
0	execcmd	Execute the command. Value: Description 0: NOACTION 1: EXECUTE	WO	0x0

flashcmdaddr

Module Instance	Base Address	Register Address
i_qspi_qspiregs	0xFF809000	0xFF809094

Offset: 0x94

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addr RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addr RW 0x0															

flashcmdaddr Fields

Bit	Name	Description	Access	Reset
31:0	addr	This should be setup before triggering the command with execute field (bit 0) of the Flash Command Control register. It is the address used by the command specified in the opcode field (bits 31:24) of the Flash Command Control register.	RW	0x0

flashcmdrddatalo

Module Instance	Base Address	Register Address
i_qspi_qspiregs	0xFF809000	0xFF8090A0

Offset: 0xA0

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
data RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
data RW 0x0															

flashcmdrddatalo Fields

Bit	Name	Description	Access	Reset
31:0	data	This is the data that is returned by the flash device for any status or configuration read operation carried out by triggering the event in the control register. The register will be valid when the polling bit in the control register is low.	RW	0x0

flashcmdrddataup

Device Instruction Configuration Register

Module Instance	Base Address	Register Address
i_qspi_qspiregs	0xFF809000	0xFF8090A4

Offset: 0xA4

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
data RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
data RW 0x0															

flashcmdrddataup Fields

Bit	Name	Description	Access	Reset
31:0	data	This is the data that is returned by the FLASH device for any status or configuration read operation carried out by triggering the event in the control register. The register will be valid when the polling bit in the control register is low.	RW	0x0

flashcmdwrdatalo

Module Instance	Base Address	Register Address
i_qspi_qspiregs	0xFF809000	0xFF8090A8

Offset: 0xA8

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
data RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
data RW 0x0															

flashcmdwrdatalo Fields

Bit	Name	Description	Access	Reset
31:0	data	This is the command write data lower byte. This should be setup before triggering the command with execute field (bit 0) of the Flash Command Control register. It is the data that is to be written to the flash for any status or configuration write operation carried out by triggering the event in the Flash Command Control register.	RW	0x0

flashcmdwrdataup

Module Instance	Base Address	Register Address
i_qspi_qspiregs	0xFF809000	0xFF8090AC

Offset: 0xAC

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
data RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
data RW 0x0															

flashcmdwrdataup Fields

Bit	Name	Description	Access	Reset
31:0	data	This is the command write data upper byte. This should be setup before triggering the command with execute field (bit 0) of the Flash Command Control register. It is the data that is to be written to the flash for any status or configuration write operation carried out by triggering the event in the Flash Command Control register.	RW	0x0

moduleid

Module Instance	Base Address	Register Address
i_qspi_qspiregs	0xFF809000	0xFF8090FC

Offset: 0xFC

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
mod_id_resv_fld RO 0x0							value RO 0x1001								
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
value RO 0x1001															

moduleid Fields

Bit	Name	Description	Access	Reset
31:25	mod_id_resv_fld		RO	0x0
24:0	value		RO	0x1001

qspi_ecc Address Map

Module Instance	Base Address	End Address
ecc_qspi_ecc_registerBlock	0xFF8C8400	0xFF8C87FF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
IP_REV_ID on page 11-315	0x0	32	RO	0x0	
CTRL on page 11-316	0x8	32	RW	0x0	ECC Control Register
INITSTAT on page 11-316	0xC	32	RW	0x0	This bit is used to set the initialize the memory and ecc to a known value
ERRINTEN on page 11-317	0x10	32	RW	0x0	Error Interrupt enable
ERRINTENS on page 11-318	0x14	32	RW	0x0	Error Interrupt set
ERRINTENR on page 11-319	0x18	32	RW	0x0	Error Interrupt reset.
INTMODE on page 11-320	0x1C	32	RW	0x0	Reads reflect SERRINTEN.
INTSTAT on page 11-320	0x20	32	RW	0x0	This bit is used to enable interrupt generation on SERR lookup table overflow. When all the entries in the table are valid=1 and this is bit is enabled, serr_req signal will be asserted.
INTTEST on page 11-321	0x24	32	RW	0x0	This bits is used to test interrupt from ECC RAM to GIC
MODSTAT on page 11-322	0x28	32	RW	0x0	Counter feature status flag

Register	Offset	Width	Access	Reset Value	Description
DERRADDRA on page 11-323	0x2C	32	RO	0x0	This register shows the address of PORTA current double-bit error. RAM size will determine the maximum number of address bits.
SERRADDRA on page 11-323	0x30	32	RO	0x0	This register shows the address of PORTA current single-bit error. RAM size will determine the maximum number of address bits.
SERRCNTREG on page 11-324	0x3C	32	RW	0x0	Maximum counter value for single-bit error interrupt
ECC_Addrbus on page 11-325	0x40	32	RW	0x0	MSB bit of address is determined by ADR.
ECC_RData0bus on page 11-326	0x44	32	RO	0x0	Data will be read to this register field.
ECC_RData1bus on page 11-326	0x48	32	RO	0x0	Data will be read to this register field.
ECC_RData2bus on page 11-327	0x4C	32	RO	0x0	Data will be read to this register field.
ECC_RData3bus on page 11-328	0x50	32	RO	0x0	Data will be read to this register field.
ECC_WData0bus on page 11-328	0x54	32	WO	0x0	Data from the register will be written to the RAM.
ECC_WData1bus on page 11-329	0x58	32	WO	0x0	Data from the register will be written to the RAM.
ECC_WData2bus on page 11-330	0x5C	32	WO	0x0	Data from the register will be written to the RAM.
ECC_WData3bus on page 11-330	0x60	32	WO	0x0	Data from the register will be written to the RAM.

Register	Offset	Width	Access	Reset Value	Description
ECC_RDataecc0bus on page 11-331	0x64	32	RO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.
ECC_RDataecc1bus on page 11-332	0x68	32	RO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.
ECC_WDataecc0bus on page 11-333	0x6C	32	WO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.
ECC_WDataecc1bus on page 11-334	0x70	32	WO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.
ECC_dbytectrl on page 11-335	0x74	32	RW	0x0	Max number of implemented byte enabled is DAT/8
ECC_accctrl on page 11-336	0x78	32	RW	0x0	These bits determine which byte of data/ecc to write to RAM.
ECC_startacc on page 11-336	0x7C	32	RW	0x0	These bits determine which byte of data/ecc to write to RAM.
ECC_wdctrl on page 11-337	0x80	32	RW	0x0	Bits to Enable/Disable Watch Dog Timer
SERRLKUPA0 on page 11-338	0x90	32	RW	0x0	Single-bit error address in LOOKUP TABLE for PORTA.

qspi_ecc Summary

Base Address: 0xFF8C8400

Register Address Offset	Bit Fields															
ecc_qspi_ecc_register-Block																
IP_REV_ID 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SIREV RO 0x0															
CTRL 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															INITA 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							CNT_RSTA 0x0	Reserved							ECC_EN 0x0
INITSTAT 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															INITCOM-PLETEA 0x0
ERRINTEN 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTEN 0x0
ERRINTENS 0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTS 0x0

Register Address Offset	Bit Fields															
ERRINTENR 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTR 0x0
INTMODE 0x1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															INTONCMP 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							INTONOVF 0x0	Reserved							INTMODE 0x0
INTSTAT 0x20	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							DERRPENA 0x0	Reserved							SERRPENA 0x0
INTTEST 0x24	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							TDERRA 0x0	Reserved							TSERRA 0x0
MODSTAT 0x28	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															CMPFLGA 0x0
DERRADDRA 0x2C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved									Address 0x0						

Register Address Offset	Bit Fields															
SERRADDRA 0x30	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved									Address 0x0						
SERRCNTREG 0x3C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	SERRCNT 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SERRCNT 0x0															
ECC_Addrbus 0x40	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved									ECC_AddrBUS 0x0						
ECC_RData0bus 0x44	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_RData1bus 0x48	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_RData2bus 0x4C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															

Register Address Offset	Bit Fields															
ECC_RData3bus 0x50	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_WData0bus 0x54	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_WData1bus 0x58	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_WData2bus 0x5C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_WData3bus 0x60	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_RDataecc0bus 0x64	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_RDataecc3BUS 0x0							Reserved	ECC_RDataecc2BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_RDataecc1BUS 0x0							Reserved	ECC_RDataecc0BUS 0x0						

Register Address Offset	Bit Fields															
ECC_RDataecc1bus 0x68	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_RDataecc7BUS 0x0							Reserved	ECC_RDataecc6BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_RDataecc5BUS 0x0							Reserved	ECC_RDataecc4BUS 0x0						
ECC_WDataecc0bus 0x6C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_WDataecc3BUS 0x0							Reserved	ECC_WDataecc2BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_WDataecc1BUS 0x0							Reserved	ECC_WDataecc0BUS 0x0						
ECC_WDataecc1bus 0x70	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_WDataecc7BUS 0x0							Reserved	ECC_WDataecc6BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_WDataecc5BUS 0x0							Reserved	ECC_WDataecc4BUS 0x0						
ECC_dbytectrl 0x74	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															DBEN 0x0
ECC_accctrl 0x78	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							RDWR 0x0	Reserved						ECCOVR 0x0	DATAOVR 0x0
ECC_startacc 0x7C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															ENBUSA 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															

Register Address Offset	Bit Fields															
ECC_wdctrl 0x80	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															WDEN_RAM 0x0
SERRLKUPA0 0x90	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	VALID RW 0x0	Reserved														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved									Address RO 0x0						

IP_REV_ID

Module Instance	Base Address	Register Address
ecc_qspi_ecc_registerBlock	0xFF8C8400	0xFF8C8400

Offset: 0x0

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SIREV RO 0x0															

IP_REV_ID Fields

Bit	Name	Description	Access	Reset
15:0	SIREV	IP Rev # These bits indicate the silicon revision number.	RO	0x0

CTRL

ECC Control Register

Module Instance	Base Address	Register Address
ecc_qspi_ecc_registerBlock	0xFF8C8400	0xFF8C8408

Offset: 0x8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															INITA 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							CNT_RSTA 0x0	Reserved							ECC_EN 0x0

CTRL Fields

Bit	Name	Description	Access	Reset
16	INITA	Enable for the hardware memory initialization PORTA.	RW	0x0
8	CNT_RSTA	Enable to reset internal single-bit error counter A value to zero	RW	0x0
0	ECC_EN	Enable for the ECC detection and correction logic.	RW	0x0

INITSTAT

This bit is used to set the initialize the memory and ecc to a known value

Module Instance	Base Address	Register Address
ecc_qspi_ecc_registerBlock	0xFF8C8400	0xFF8C840C

Offset: 0xC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															INITCOM-PLETEA 0x0

INITSTAT Fields

Bit	Name	Description	Access	Reset
0	INITCOMPLETEA	This bit is used to verify if the hardware memory initialization has completed PORTB.	RW	0x0

ERRINTEN

Error Interrupt enable

Module Instance	Base Address	Register Address
ecc_qspi_ecc_registerBlock	0xFF8C8400	0xFF8C8410

Offset: 0x10

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															SERRINTEN 0x0

ERRINTEN Fields

Bit	Name	Description	Access	Reset
0	SERRINTEN	This bit is used to enable the single bit error interrupt of ECC RAM system	RW	0x0

ERRINTENS

Error Interrupt set

Module Instance	Base Address	Register Address
ecc_qspi_ecc_registerBlock	0xFF8C8400	0xFF8C8414

Offset: 0x14

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															SERRINTS 0x0

ERRINTENS Fields

Bit	Name	Description	Access	Reset
0	SERRINTS	This bit is used to set the single-bit error interrupt bit.	RW	0x0

ERRINTENR

Error Interrupt reset.

Module Instance	Base Address	Register Address
ecc_qspi_ecc_registerBlock	0xFF8C8400	0xFF8C8418

Offset: 0x18

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															SERRINTR 0x0

ERRINTENR Fields

Bit	Name	Description	Access	Reset
0	SERRINTR	This bit is used to reset the single-bit error interrupt bit. Reads reflect SERRINTEN. 1'b0: Writing of zero has no effect. 1'b1: By writing one, this bit will reset SERRINTEN bit to 0. This is performing a bitwise writing of this feature.	RW	0x0

INTMODE

Reads reflect SERRINTEN.

Module Instance	Base Address	Register Address
ecc_qspi_ecc_registerBlock	0xFF8C8400	0xFF8C841C

Offset: 0x1C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															INTONCMP 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							INTONOVF 0x0	Reserved							INTMODE 0x0

INTMODE Fields

Bit	Name	Description	Access	Reset
16	INTONCMP	Enable interrupt on compare.	RW	0x0
8	INTONOVF	Enable interrupt on overflow.	RW	0x0
0	INTMODE	Interrupt mode for single-bit errors.	RW	0x0

INTSTAT

This bit is used to enable interrupt generation on SERR lookup table overflow. When all the entries in the table are valid=1 and this is bit is enabled, serr_req signal will be asserted.

Module Instance	Base Address	Register Address
ecc_qspi_ecc_registerBlock	0xFF8C8400	0xFF8C8420

Offset: 0x20

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							DERRPENA 0x0	Reserved							SERRPENA 0x0

INTSTAT Fields

Bit	Name	Description	Access	Reset
8	DERRPENA	Double-bit error pending for PORTA.	RW	0x0
0	SERRPENA	Single-bit error pending for PORTA.	RW	0x0

INTTEST

This bits is used to test interrupt from ECC RAM to GIC

Module Instance	Base Address	Register Address
ecc_qspi_ecc_registerBlock	0xFF8C8400	0xFF8C8424

Offset: 0x24

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							TDERRA 0x0	Reserved							TSERRA 0x0

INTTEST Fields

Bit	Name	Description	Access	Reset
8	TDERRA	Test PORTA Double-bit error.	RW	0x0
0	TSERRA	Test PORTA Single-bit error.	RW	0x0

MODSTAT

Counter feature status flag

Module Instance	Base Address	Register Address
ecc_qspi_ecc_registerBlock	0xFF8C8400	0xFF8C8428

Offset: 0x28

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															CMPFLGA 0x0

MODSTAT Fields

Bit	Name	Description	Access	Reset
0	CMPFLGA	Port A compare status flag	RW	0x0

DERRADDRA

This register shows the address of PORTA current double-bit error. RAM size will determine the maximum number of address bits.

Module Instance	Base Address	Register Address
ecc_qspi_ecc_registerBlock	0xFF8C8400	0xFF8C842C

Offset: 0x2C

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									Address 0x0						

DERRADDRA Fields

Bit	Name	Description	Access	Reset
6:0	Address	Recent double-bit error address.	RO	0x0

SERRADDRA

This register shows the address of PORTA current single-bit error. RAM size will determine the maximum number of address bits.

Module Instance	Base Address	Register Address
ecc_qspi_ecc_registerBlock	0xFF8C8400	0xFF8C8430

Offset: 0x30

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									Address 0x0						

SERRADDRA Fields

Bit	Name	Description	Access	Reset
6:0	Address	Recent single-bit error address.	RO	0x0

SERRCNTREG

Maximum counter value for single-bit error interrupt

Module Instance	Base Address	Register Address
ecc_qspi_ecc_registerBlock	0xFF8C8400	0xFF8C843C

Offset: 0x3C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SERRCNT 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SERRCNT 0x0															

SERRCNTREG Fields

Bit	Name	Description	Access	Reset
31:0	SERRCNT	Counter value	RW	0x0

ECC_Addrbus

MSB bit of address is determined by ADR.

Module Instance	Base Address	Register Address
ecc_qspi_ecc_registerBlock	0xFF8C8400	0xFF8C8440

Offset: 0x40

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									ECC_AddrBUS 0x0						

ECC_Addrbus Fields

Bit	Name	Description	Access	Reset
6:0	ECC_AddrBUS	Address will be driven to RAM to either read or write the data. Address will be latched by the RAM when the Enbus is asserted.	RW	0x0

ECC_RData0bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_qspi_ecc_registerBlock	0xFF8C8400	0xFF8C8444

Offset: 0x44

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData0bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_RDataBUS	ECC_RDataBUS[31:0].	RO	0x0

ECC_RData1bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_qspi_ecc_registerBlock	0xFF8C8400	0xFF8C8448

Offset: 0x48

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData1bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_RDataBUS	ECC_RDataBUS[63:32].	RO	0x0

ECC_RData2bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_qspi_ecc_registerBlock	0xFF8C8400	0xFF8C844C

Offset: 0x4C

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData2bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_RDataBUS	ECC_RDataBUS[95:64].	RO	0x0

ECC_RData3bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_qspi_ecc_registerBlock	0xFF8C8400	0xFF8C8450

Offset: 0x50

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData3bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_RDataBUS	ECC_RDataBUS[127-96].	RO	0x0

ECC_WData0bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_qspi_ecc_registerBlock	0xFF8C8400	0xFF8C8454

Offset: 0x54

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData0bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_WDataBUS	ECC_WDataBUS[31:0].	WO	0x0

ECC_WData1bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_qspi_ecc_registerBlock	0xFF8C8400	0xFF8C8458

Offset: 0x58

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData1bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_WDataBUS	ECC_WDataBUS[63:32].	WO	0x0

ECC_WData2bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_qspi_ecc_registerBlock	0xFF8C8400	0xFF8C845C

Offset: 0x5C

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData2bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_WDataBUS	ECC_WDataBUS[95-64].	WO	0x0

ECC_WData3bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_qspi_ecc_registerBlock	0xFF8C8400	0xFF8C8460

Offset: 0x60

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData3bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_WDataBUS	ECC_WDataBUS[127-96].	WO	0x0

ECC_RDataecc0bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_qspi_ecc_registerBlock	0xFF8C8400	0xFF8C8464

Offset: 0x64

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	ECC_RDataecc3BUS 0x0							Reserved	ECC_RDataecc2BUS 0x0						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	ECC_RDataecc1BUS 0x0							Reserved	ECC_RDataecc0BUS 0x0						

ECC_RDataecc0bus Fields

Bit	Name	Description	Access	Reset
30:24	ECC_RDataecc3BUS	Eccdata will be read to this register field.	RO	0x0
22:16	ECC_RDataecc2BUS	Eccdata will be read to this register field.	RO	0x0
14:8	ECC_RDataecc1BUS	Eccdata will be read to this register field.	RO	0x0
6:0	ECC_RDataecc0BUS	Eccdata will be read to this register field.	RO	0x0

ECC_RDataecc1bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_qspi_ecc_registerBlock	0xFF8C8400	0xFF8C8468

Offset: 0x68

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	ECC_RDataecc7BUS 0x0							Reserved	ECC_RDataecc6BUS 0x0						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	ECC_RDataecc5BUS 0x0							Reserved	ECC_RDataecc4BUS 0x0						

ECC_RDataecc1bus Fields

Bit	Name	Description	Access	Reset
30:24	ECC_RDataecc7BUS	Eccdata will be read to this register field.	RO	0x0
22:16	ECC_RDataecc6BUS	Eccdata will be read to this register field.	RO	0x0
14:8	ECC_RDataecc5BUS	Eccdata will be read to this register field.	RO	0x0
6:0	ECC_RDataecc4BUS	Eccdata will be read to this register field.	RO	0x0

ECC_WDataecc0bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_qspi_ecc_registerBlock	0xFF8C8400	0xFF8C846C

Offset: 0x6C

Access: WO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	ECC_WDataecc3BUS 0x0							Reserved	ECC_WDataecc2BUS 0x0						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	ECC_WDataecc1BUS 0x0							Reserved	ECC_WDataecc0BUS 0x0						

ECC_WDataecc0bus Fields

Bit	Name	Description	Access	Reset
30:24	ECC_WDataecc3BUS	Eccdata from the register will be written to the RAM.	WO	0x0
22:16	ECC_WDataecc2BUS	Eccdata from the register will be written to the RAM.	WO	0x0
14:8	ECC_WDataecc1BUS	Eccdata from the register will be written to the RAM.	WO	0x0
6:0	ECC_WDataecc0BUS	Eccdata from the register will be written to the RAM.	WO	0x0

ECC_WDataecc1bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_qspi_ecc_registerBlock	0xFF8C8400	0xFF8C8470

Offset: 0x70

Access: WO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	ECC_WDataecc7BUS 0x0							Reserved	ECC_WDataecc6BUS 0x0						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	ECC_WDataecc5BUS 0x0							Reserved	ECC_WDataecc4BUS 0x0						

ECC_WDataecc1bus Fields

Bit	Name	Description	Access	Reset
30:24	ECC_WDataecc7BUS	Eccdata from the register will be written to the RAM.	WO	0x0
22:16	ECC_WDataecc6BUS	Eccdata from the register will be written to the RAM.	WO	0x0
14:8	ECC_WDataecc5BUS	Eccdata from the register will be written to the RAM.	WO	0x0
6:0	ECC_WDataecc4BUS	Eccdata from the register will be written to the RAM.	WO	0x0

ECC_dbytectrl

ECC Data Byte Control. The maximum number of implemented byte enables equals the data width of the RAM divided by 8.

Module Instance	Base Address	Register Address
ecc_qspi_ecc_registerBlock	0xFF8C8400	0xFF8C8474

Offset: 0x74

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															DBEN 0x0

ECC_dbytectrl Fields

Bit	Name	Description	Access	Reset
0	DBEN	Byte or word enable for access.	RW	0x0

ECC_accctrl

These bits determine which byte of data/ecc to write to RAM.

Module Instance	Base Address	Register Address
ecc_qspi_ecc_registerBlock	0xFF8C8400	0xFF8C8478

Offset: 0x78

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							RDWR 0x0	Reserved						ECCOVR 0x0	DATAOVR 0x0

ECC_accctrl Fields

Bit	Name	Description	Access	Reset
8	RDWR	Control for read/write.	RW	0x0
1	ECCOVR	ECC Data Override.	RW	0x0
0	DATAOVR	RAM Data Override. Override the ECC_dataBUS register with RAM data in read mode set by ECC_RW. 1'b0: Data override disabled. 1'b1: Data override enabled.	RW	0x0

ECC_startacc

These bits determine which byte of data/ecc to write to RAM.

Module Instance	Base Address	Register Address
ecc_qspi_ecc_registerBlock	0xFF8C8400	0xFF8C847C

Offset: 0x7C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															ENBUSA 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															

ECC_startacc Fields

Bit	Name	Description	Access	Reset
16	ENBUSA	Start RAM access for PORTA.	RW	0x0

ECC_wdctrl

Bits to Enable/Disable Watch Dog Timer

Module Instance	Base Address	Register Address
ecc_qspi_ecc_registerBlock	0xFF8C8400	0xFF8C8480

Offset: 0x80

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															WDEN_RAM 0x0

ECC_wdctrl Fields

Bit	Name	Description	Access	Reset
0	WDEN_RAM	Enable watchdog timeout for OCP register access to IP RAM.	RW	0x0

SERRLKUPA0

Single-bit error address in LOOKUP TABLE for PORTA.

Module Instance	Base Address	Register Address
ecc_qspi_ecc_registerBlock	0xFF8C8400	0xFF8C8490

Offset: 0x90

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VALID RW 0x0	Reserved														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									Address RO 0x0						

SERRLKUPA0 Fields

Bit	Name	Description	Access	Reset
31	VALID	Valid flag bit. Valid bit indicates if the address in this register is current or stale.	RW	0x0
6:0	Address	Recent Single-bit error address. This register shows the address of the each single-bit error. RAM size will determine the maximum number of address bits. If ram size is 32 Kbytes, bit 30-16 will be reserved and read as zero.	RO	0x0

Document Revision History

Table 15-5: Document Revision History

Date	Version	Changes
October 2016	2016.10.28	Maintenance release
May 2016	2016.05.27	• Changed the name of the internal QSPI reference clock from qspi_clk to qspi_ref_clk; and the external QSPI output clock, from sclk_out to qspi_clk. • Added a link to the *Supported Flash Devices for Arria 10 SoC* webpage. • Re-worded information about disabling the watermark feature in the "Indirect Read Operation" and "Indirect Write Operation" sections.
May 2016	2016.05.03	Maintenance release
November 2015	2015.11.02	• Renamed "Interface Pins" section to "Quad SPI Flash Controller Signal Description" and moved it below the "Quad SPI Flash Controller Block Diagram and System Integration" section • Corrected the link to the HPS Address Map. • Added the Arria 10 register map. • Better defined l4_main_clk clock. • Added Clock Gating information.
May 2015	2015.05.04	Added information about clearing out the ECC before the feature is enabled
December 2014	2014.12.15	Maintenance release
August 2014	2014.08.18	Initial release

DMA Controller 16

2016.10.28

a10_5v4 Subscribe Send Feedback

This chapter describes the direct memory access controller (DMAC) contained in the hard processor system (HPS). The DMAC transfers data between memory and peripherals and other memory locations in the system. The DMA controller is an instance of the ARM Corelink DMA Controller (DMA-330).

- Microcoded to support flexible transfer types
- Supports up to eight channels
- Provides 8-, 16-, 32-, and 64-bit transfer support
- Supports flow control with 32 peripheral request interfaces

Related Information

ARM Information Center

For more information about ARM's DMA-330 controller, refer to the *CoreLink DMA Controller DMA-330 Revision: r1p2* Technical Reference Manual on the ARM Infocenter website.

Features of the DMA Controller

The HPS provides one DMAC to handle the data transfer between memory-mapped peripherals and memories, off-loading this work from the MPU subsystem.

- The DMAC supports multiple transfer types:
  - Memory-to-memory
  - Memory-to-peripheral
  - Peripheral-to-memory
  - Scatter-gather
- Supports up to eight DMA channels
- Supports up to eight outstanding AXI read and eight outstanding AXI write transactions
- Supports scheduling up to 16 outstanding read and 16 outstanding write instructions
- Supports nine interrupt lines into the MPU subsystem:
  - One for DMA thread abort
  - Eight for events
- Supports 32 peripheral request interfaces:
  - Eight for FPGA
    - FPGA 5 is multiplexed with Security Manager TX
    - FPGA 6 is multiplexed with I^2C_EMAC2_TX
    - FPGA 7 is multiplexed with I^2C_EMAC2_RX
  - Five for I^2C
  - Three for I^2C (EMAC)
  - Eight for SPI
  - Two for quad SPI
  - One for System Trace Macrocell
  - Four for UART
  - One for FPGA manager

The DMA controller provides:

- An instruction processing block that enables it to process program code that controls a DMA transfer
- An ARM Advanced Microcontroller Bus Architecture (AMBA) Advanced eXtensible Interface (AXI) master interface unit to fetch the program code from system memory into its instruction cache

  Note: The AXI master interface is used to perform DMA data transfer as well. The DMA instruction execution engine executes the program code from its instruction cache and schedules read or write AXI instructions through the respective instruction queues.
- A multi-FIFO (MFIFO) data buffer that it uses to store data that it reads, or writes, during a DMA transfer
- Nine interrupt outputs to enable efficient communication of events to the MPU interrupt controller

  Note: The peripheral request interfaces support the connection of DMA-capable peripherals to enable memory-to-peripheral and peripheral-to-memory transfers to occur, without intervention from the processor. Since the HPS supports some peripherals that do not comply with ARM DMA peripheral interface protocol, adapters are added to allow these peripherals to work with the DMAC.

The following peripheral interface protocols are supported:

- Synopsys protocol
  - FPGA manager
  - Serial peripheral interface (SPI)
  - Universal asynchronous receiver transmitter (UART)
  - Inter-integrated circuit (I^2C)
  - FPGA
- ARM protocol
  - Quad SPI flash controller
  - System trace macrocell (STM)

Dual slave interfaces enable the operation of the DMA controller to be partitioned into a secure and non-secure state. The network interconnect must be configured to ensure that only secure transactions can access the secure interface. The slave interfaces can access status registers and also be used to directly issue and execute instructions in the DMA controller.

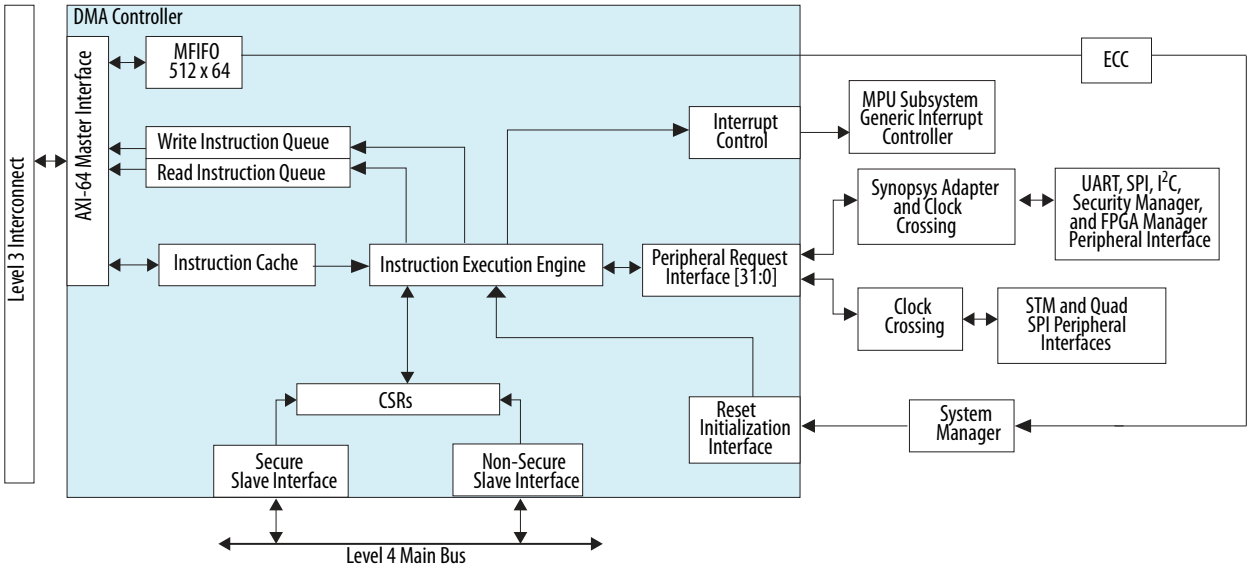
The DMAC has the following features:

- A small instruction set that provides a flexible method of specifying the DMA operations. This architecture provides greater flexibility than the fixed capabilities of a Linked-List Item (LLI) based DMA controller

DMA Controller Block Diagram and System Integration

The following figure shows a block diagram of the DMAC and how it integrates into the rest of the HPS system.

Figure 16-1: DMA Controller Connectivity



The l4_main_clk clock drives the DMA controller, controller logic, and all the interfaces. The DMA controller accesses the level 3 (L3) main switch with its 64-bit AXI master interface.

The DMA controller provides the following slave interfaces that connect to the L4 bus:

- Non-secure slave interface
- Secure slave interface
- ECC register slave interface

Both non-secure and secure slave interfaces may access registers that control the functionality of the DMA controller. The DMA controller implements TrustZone secure technology with one interface operating in the secure state and the other operating in the non-secure state.

The MFIFO has an ECC controller built-in to provide ECC protection. The ECC controller is able to detect single-bit and double-bit errors, and correct the single-bit errors. The ECC operation and functionality is programmable via the ECC register slave interface, as shown in **Figure 16-1**. The ECC register interface provides host access to configure the ECC logic as well as inject bit errors into the memory. It also provides host access to memory initialization hardware used to clear out the memory contents including the ECC bits. The ECC controller generates interrupts upon occurrences of single and double-bit errors, and the interrupt signals are connected to the system manager.

Related Information

Error Checking and Correction Controller on page 11-1
Error Checking and Correction (ECC) controllers provide single- and double-bit error memory protection for integrated on-chip RAM and peripheral RAMs within the hard processor system (HPS).

Functional Description of the DMA Controller

This section describes the major interfaces and components of the DMAC and its operation.

The DMAC contains an instruction processing block that processes program code to control a DMA transfer. The program code is stored in a region of system memory that the DMAC accesses using its AXI master interface. The DMAC stores instructions temporarily in an internal cache.

The DMAC has eight DMA channels. Each channel supports a single concurrent thread of DMA operation. In addition, a single DMA manager thread exists, and you can use it to initialize the DMA channel threads.

The DMAC executes one instruction per clock cycle. To ensure that it regularly executes each active thread, the DMAC alternates by processing the DMA manager thread and then a DMA channel thread. It performs a round-robin process when selecting the next active DMA channel thread to execute.

The DMAC uses variable-length instructions that consist of one to six bytes. It provides a separate program counter (PC) register for each DMA channel.

The DMAC includes a 16-line instruction cache to improve the instruction fetch performance. Each instruction cache line contains eight, four-byte words for a total cache line size of 32 bytes. The DMAC instruction cache size is therefore 16 lines times 32 bytes per line which equals 512 bytes. When a thread requests an instruction from an address, the cache performs a lookup. If a cache hit occurs, then the cache immediately provides the instruction. Otherwise, the thread is stalled while the DMAC performs a cache line fill through the AXI master interface. If an instruction spans the end of a cache line, the DMAC performs multiple cache accesses to fetch the instruction.

Note: When a cache line fill is in progress, the DMAC enables other threads to access the cache. But if another cache miss occurs, the pipeline stalls until the first line fill is complete.

When a DMA channel thread executes a load or store instruction, the DMAC adds the instruction to the relevant read or write queue. The DMAC uses these queues as an instruction storage buffer prior to it issuing the instructions on the AXI. The DMAC also contains an MFIFO data buffer in which it stores data that it reads or writes during a DMA transfer.

The DMAC provides multiple interrupt outputs to enable efficient communication of events to the system CPUs. The peripheral request interfaces support the connection of DMA-capable peripherals to enable memory-to-peripheral and peripheral-to-memory DMA transfers to occur without intervention from the microprocessor.

Dual slave interfaces enable the operation of the DMAC to be partitioned into the secure state and non-secure states. You can access status registers and also directly execute instructions in the DMAC with the slave interfaces.

AXI Characteristics for a DMA Transfer

This table is a reference table to find out more information about how the DMAC AXI controls the AXI control signals.

Table 16-1: AXI Characteristics for a DMA Transfer

<table>
<tr><th>Access Type</th><th>Protection</th><th>Length</th><th>Burst</th><th>Size</th><th>Cache</th></tr>
<tr><td>DMA channel load</td><td colspan="5" rowspan="2">For more information, refer to the "Channel Control Registers" chapter in the CoreLink DMA Controller DMA-330 Revision: r1p2 Technical Reference Manual.</td></tr>
<tr><td>DMA channel store</td></tr>
<tr><td>DMA manager instruction fetch</td><td>Privileged. Secure state from DNS(45) bit.</td><td rowspan="2">See ARLEN and ARSIZE for instruction fetches.</td><td rowspan="2">INCR</td><td rowspan="2">See ARLEN and ARSIZE for instruction fetches.</td><td rowspan="2">• Cacheable write-through
• Allocate on reads only</td></tr>
<tr><td>DMA channel instruction fetch</td><td>Privileged. Secure state from CNS(46) bit.</td></tr>
</table>

Note: Any cacheable access made by the DMA master is routed to the ACP.

(45) The DSR Register contains the DNS bit. For more information, refer to the "DMA Manager Status Register" chapter in the *CoreLink DMA Controller DMA-330 Revision: r1p2* Technical Reference Manual.

(46) The CSR_n Register contains the CNS bit for DMA channel <*n*>. For more information, refer to the "Channel Status Registers" chapter in the *CoreLink DMA Controller DMA-330 Revision: r1p2* Technical Reference Manual.

ARLEN and ARSIZE for Instruction Fetches

When performing an instruction fetch, the DMAC sets **ARLEN** and **ARSIZE** as follows:

- Instruction cache length ≤ AXI data bus width
  - **ARLEN** = 1
  - **ARSIZE** = length of instruction cache in bytes
- Instruction cache length > AXI data bus width
  - **ARLEN** = ratio of the length of an instruction cache line in bytes to the width of the AXI data bus in bytes
  - **ARSIZE** = width of AXI data bus in bytes

Related Information

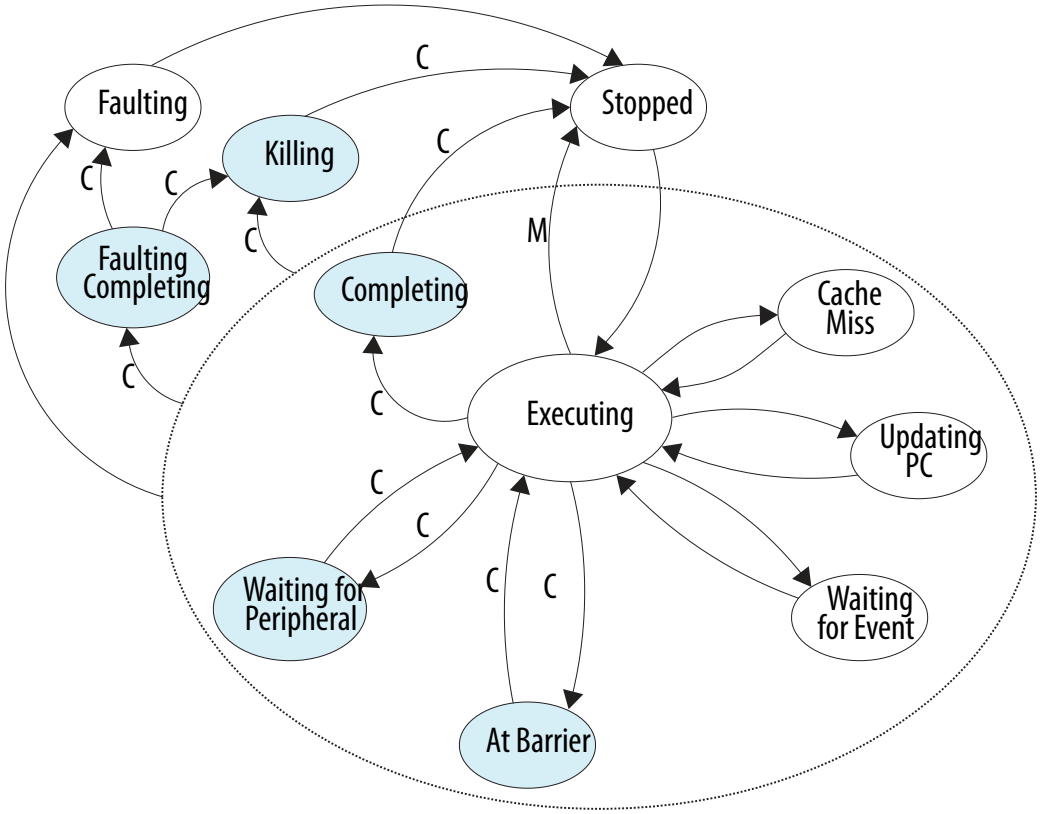
ARM Information Center

For more information about ARM's DMA-330 controller, refer to the *CoreLink DMA Controller DMA-330 Revision: r1p2* Technical Reference Manual on the ARM Infocenter website.

Operating States

The following figure shows the transitions among operating states for the DMA manager thread and DMA channel threads. The DMAC provides a separate state machine for each thread.

Figure 16-2: Thread Operating States



In the above figure, the DMAC permits all of the following:

- Only DMA channel threads can use states shown in blue
- Arcs with no letter designator indicate state transitions for the DMA manager and DMA channel threads, otherwise use is restricted as follows:
  - **C**—DMA channel threads only
  - **M**—DMA manager thread only
- States within the dotted line can transition to the Faulting completing, Faulting, or Killing

After the DMAC exits from reset, it sets all DMA channel threads to the Stopped state, and the status of `boot_from_pc` controls the DMA manager thread state.

- When `boot_from_pc` is LOW, then the DMA manager thread moves to the Stopped state.
- When `boot_from_pc` is HIGH, then the DMA manager thread moves to the Executing state.

The following list describes the Thread Operating states:

Stopped

The thread has an invalid PC (Program Counter) and it is not fetching instructions. Depending on the thread type, you can cause the thread to move to the Executing state by all of the following:

- DMA manager thread—Issuing the DMAGO instruction through the slave interface
- DMA channel thread—Programming the DMA manager thread to execute `DMAGO` for a DMA channel thread in the Stopped state

Executing

The thread has a valid PC and therefore the DMAC includes the thread when it arbitrates. The thread can then change to one of the following states under the following conditions:

- **Stopped**—When the DMA manager thread executes `DMAEND`
- **Cache Miss**—When the instruction cache does not contain the next instruction for either the DMA manager thread or the DMA channel thread
- **Updating PC**—When the DMAC calculates the address of the next access in the cache
- **Waiting for Event**—When a thread executes `DMAWFE`
- **At Barrier**—When a DMA channel thread either:
  - Executes `DMARMB`, `DMAWMB`, or `DMAFLUSHP`
  - Updates control registers that affect alignment during a DMA Cycle
- **Waiting for Peripheral**—When a DMA channel thread executes `DMAWFP`
- **Killing**—When a DMA channel thread executes `DMAKILL`
- **Faulting Completing**—For a DMA channel thread when either:
  - The thread executes an undefined or invalid instruction
  - An AXI error occurs during an instruction fetch or data transfer
- **Faulting**—For the DMA manager thread when either:
  - The thread executes an undefined or invalid instruction
  - An AXI error occurs during an instruction fetch

  For a DMA channel thread when a watchdog timeout abort occurs.
- **Completing**—When a DMA channel thread executes `DMAEND`

Cache Miss

The thread is stalled and the DMAC is performing a cache line fill. After it completes the cache fill, the thread returns to the Executing state.

Updating PC

The DMAC is calculating the address of the next access in the cache. After it calculates the PC, the thread returns to the Executing state.

Waiting For Event

The thread is stalled and is waiting for the DMAC to execute `DMASEV` using the corresponding event number. After the corresponding event occurs, the thread returns to the Executing state.

At Barrier

A DMA channel thread is stalled and the DMAC is waiting for transactions on the AXI to complete. After the AXI transactions complete, the thread returns to the Executing state.

Waiting For Peripheral

A DMA channel thread is stalled and the DMAC is waiting for the peripheral to provide the requested data. After the peripheral provides the data, the thread returns to the Executing state.

Faulting Completing

A DMA channel thread is waiting for the AXI master interface to signal that the outstanding load or store transactions are complete. After the transactions complete, the thread moves to the Faulting state.

Faulting

The thread is stalled indefinitely. The thread moves to the Stopped state when you use the `DBGCMD` register to instruct the DMAC to execute `DMAKILL` for that thread.

Killing

A DMA channel thread is waiting for the AXI master interface to signal that the outstanding load or store transactions are complete. After the transactions complete, the thread moves to the Stopped state.

Completing

A DMA channel thread is waiting for the AXI master interface to signal that the outstanding load or store transactions are complete. After the transactions complete, the thread moves to the Stopped state.

Related Information

Updating DMA Channel Control Registers During a DMA Cycle on page 16-31

Error Checking and Correction

The SRAM local memory buffer is 64 by 512 bits, providing 4096 bytes of memory. The buffer provides ECC capability. The ECC block is integrated around the SRAM local memory buffer and provides the following features:

- Output to notify the system manager when single-bit correctable errors are detected and corrected
- Output to notify the system manager when double-bit uncorrectable errors are detected
- Provision for the injection of single-bit and double-bit errors for test purposes

The ECC is disabled by default. For information on using the ECC feature, refer to the ECC chapter in the Arria 10 Device handbook.

The system manager provides registers to set or mask single-bit or double-bit ECC error interrupts.

Related Information

Error Checking and Correction Controller on page 11-1
Error Checking and Correction (ECC) controllers provide single- and double-bit error memory protection for integrated on-chip RAM and peripheral RAMs within the hard processor system (HPS).

Initializing and Clearing of Memory before Enabling ECC

Due to the DMA controller FIFO implementation, you must initialize and clear the FIFO before you enable the ECC to avoid a single event upset (SEU).

The following describes the initialization requirements:

- You must write a known pattern for data and ECC syndrome bits in memory, which involves the initialization of data to zero and corresponding nonzero ECC in hardware.
- Software must wait for the initialization process to complete before memory access is allowed. The initialization process cannot be interrupted nor stopped.

Related Information

Error Checking and Correction Controller on page 11-1
For more information about ECC, refer to the *Error Checking and Correction Controller* chapter of the Arria 10 Device Handbook.

Initializing the DMAC

The DMAC provides several memory-mapped control signals that initialize its operating state when it exits from reset. The DMAC does not automatically begin executing code when it exits from reset. The system manager controls these memory-mapped control signals.

Related Information

ARM Information Center
For more information about ARM's DMA-330 controller, refer to the *CoreLink DMA Controller DMA-330 Revision: r1p2* Technical Reference Manual on the ARM Infocenter website.

How to Set the Security State of the DMA Manager

The security state of the DMAC manager thread is controlled by a bit in the DMA register in the System Manager. The `mgr_ns` bit controls the security state and when released from reset the DMA is in the secure state.

Related Information

- **DMA Manager Thread in Secure State** on page 16-28
  Describes how the security state of the DMA manager affects how the DMAC operates.
- **DMA Manager Thread in Non-Secure State** on page 16-28

How to Set the Security State for the Interrupt Outputs

You can set the security of an event-interrupt resource through the `irq_ns` bitfield in the DMA register found in the System Manager register map.

Related Information

Security Usage on page 16-27
Describes how the security state of the irq[x] signals affects how the DMAC executes the DMAWFE and DMASEV instructions.

How to Set the Security State for a Peripheral Request Interface

You can control the security state of each peripheral request interface by programming bits in the `dma_periph` register found in the System Manager. Each bit corresponds to a peripheral request interface. So if bit 0 is 0, then the peripheral interface 0 operates in the secure state.

Related Information

Security Usage on page 16-27
Describes the effect of the security state of the peripheral request interfaces on the execution of the DMAWFP, DMALDP, DMASTP, or DMAFLUSHP instructions by a DMA channel thread.

Using the Slave Interfaces

The slave interfaces connect the DMAC to the level 4 (L4) main bus and enable the MPU to access the registers. Using these registers, the MPU can perform the following functions:

- Access the status of the DMA manager thread
- Access the status of the DMA channel threads
- Enable or clear interrupts
- Enable events
- Execute an instruction for the DMAC by programming the following debug registers:
  - `DBGCMD` register
  - `DBGINST0` register
  - `DBGINST1` register

Issuing Instructions to the DMAC using a Slave Interface

When the DMAC is operating, you can only issue the following instructions:

- `DMAGO`—Starts a DMA transaction using a DMA channel that you specify
- `DMASEV`—Signals the occurrence of an event, or interrupt, using an event number that you specify
- `DMAKILL`—Terminates a thread

You must ensure that you use the appropriate slave interface, depending on the security state in which the `boot_manager_ns` signal initializes the DMAC. For example, if the DMAC is in the secure state, you must

issue the instruction using the secure slave interface, otherwise the DMAC ignores the instruction. You can use the secure or non-secure slave interface to start or restart a DMA channel when the DMAC is in the non-secure state.

Note: Before you can issue instructions using the debug instruction registers or the DBGCMD register, you must read the DBGSTATUS register to ensure that debug is idle, otherwise the DMAC ignores the instructions.

The DMAC immediately processes any instructions received from a slave interface, unless the pipeline is busy processing another instruction.

Note: Prior to issuing DMAGO, you must ensure that the system memory contains a suitable program for the DMAC to execute, starting at the address that the DMAGO specifies.

Using DMAGO with the Debug Instruction Registers

The following example shows the necessary steps to start a DMA channel thread using the debug instruction registers:

1. Create a program for the DMA channel.
2. Store the program in a region of system memory.
3. Program one of the slave interfaces on the DMAC to a DMAGO instruction as follows:
   a. Poll the DBGSTATUS register to ensure that debug is idle, and the dbgstatus bit is 0.
   b. Write to the DBGINST0 register and enter all of the following:
      - Instruction byte 0 encoding for DMAGO
      - Instruction byte 1 encoding for DMAGO
      - Debug thread bit to 0 to select the DMA manager thread
   c. Write to the DBGINST1 register with the DMAGO instruction byte [5:2] data. You must set these four bytes to the address of the first instruction in the program that is written to system memory in **step 2** above.
4. Instruct the DMAC to execute the instruction that the debug instruction registers contain by writing zero to the DBGCMD register. The DMAC starts the DMA channel thread and sets the dbgstatus bit to 1. After the DMAC completes execution of the instruction, it clears the dbgstatus bit to 0.

Peripheral Request Interface

You can enable each direct memory access (DMA) controller peripheral request interface, individually. Setting each of the eight peripheral request IDs enables or disables a peripheral request interface. When set to enable, it allows for FPGA soft logic to request a DMA transfer. For DMA transfers to or from the FPGA, this feature is only necessary if your design requires transfer flow control.

The DMA peripheral request interface communicates with peripherals by either the ARM protocol or the Synopsys protocol.

- ARM Protocol -

  For peripherals using the ARM protocol, clock-crossing logic is the only logic between the DMA and the peripheral.
- Synopsys Protocol Overview -

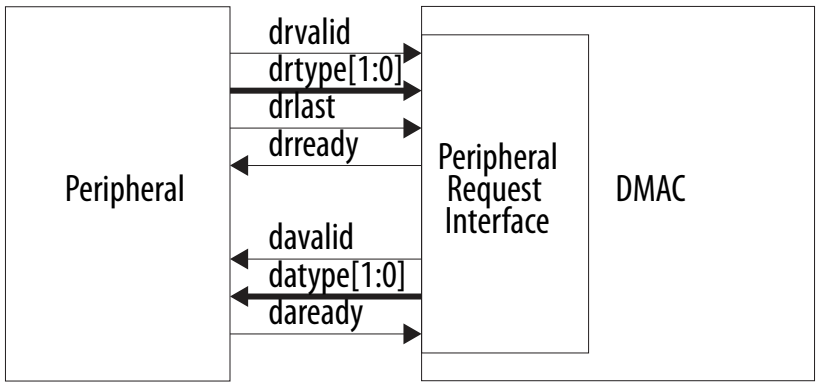
  The Synopsys protocol is used for the hardware flow control that is not the same as the protocol used by the DMA-330 core.

  For peripherals using the Synopsys protocol, clock-crossing logic and protocol adaptation logic exist between the DMA and the peripheral.

The following figure shows that the peripheral request interface consists of a peripheral request bus and a DMAC acknowledge bus that use the prefixes:

- `dr`—The peripheral request bus
- `da`—The DMAC acknowledge bus

Figure 16-3: Request and Acknowledge Buses on the Peripheral Request Interface



Both buses use the valid and ready handshake that the AXI protocol describes.

The peripheral uses `drtype[1:0]` to either:

- Request a single transfer
- Request a burst transfer
- Acknowledge a flush request

The peripheral uses `drlast` to notify the DMAC that the request on `drtype[1:0]` is the last request of the DMA transfer sequence. `drlast` is transferred at the same time as `drtype[1:0]`.

The DMAC can indicate the following using `datype[1:0]`:

- When it completes the requested single transfer
- When it completes the requested burst transfer
- When it issues a flush request

Note: If you configure the DMAC to provide more than one peripheral request interface, each interface is assigned a unique identifier, _<x> where <x> represents the number of the interface.

For the Synopsys protocol, the following signals are used in the handshaking protocol:

- dma_tx_req_n
- dma_rx_req_n
- dma_tx_ack_n
- dma_rx_ack_n
- dma_tx_single_n
- dma_rx_single_n

Handshake Rules

The DMAC uses the DMA handshake rules that are listed, below, when a DMA channel thread is active, that is, not in the **Stopped** state.

- drvalid can change from LOW to HIGH on any aclk cycle, but it must only change from HIGH to LOW when drready is HIGH.
- drtype can only change when either drready is HIGH, or drvalid is LOW.
- drlast can only change when either drready is HIGH, or drvalid is LOW.
- davalid can change from LOW to HIGH on any aclk cycle, but it must only change from HIGH to LOW when daready is HIGH.
- datype can only change when either daready is HIGH, OR davalid is LOW.

Table 16-2: DMA Peripheral Interface Signal Definition

Signal	Description
drready	Indicates whether the DMAC can accept the information that the peripheral provides on drtype_<x>[1:0]: • 0 = DMAC not ready • 1 = DMAC ready **Note:** If drvalid is HIGH then the DMAC sets drready to HIGH when it accepts the peripheral request.
drvalid	Indicates when the peripheral provides valid control information: • 0 = No control information is available • 1 = drtype_<x>[1:0] and drlast_<x> contain valid information for the DMAC **Note:** The peripheral sets drvalid HIGH when it starts to provide valid control information on drlast and drtype. The state of drvalid, drlast and drtype must remain constant until the DMAC sets drready HIGH.
drtype[1:0]	Indicates the type of acknowledgment, or request, that the peripheral signals: • b00 = single level request • b01 = burst level request • b10 = acknowledging a flush request that the DMAC requested • b11 = reserved

Signal	Description
drlast	Indicates that the peripheral is sending the last data transfer for the current DMA transfer: • 0 = last data request is not in progress • 1 = last data request is in progress **Note:** The DMAC only uses this signal when drtype_<x>[1:0] is b00 or b01.
daready	Indicates whether the peripheral can accept the information that the DMAC provides on datype_<x>[1:0]: • 0 = peripheral not ready • 1 = peripheral ready **Note:** If davalid is HIGH, the peripheral sets daready HIGH when either it: • Accepts a flush request from the DMAC • Acknowledges the completion of a DMA transfer
davalid	Indicates when the DMAC provides valid control information: • 0 = no control information is available • 1 = datype_<x>[1:0] contains valid information for the peripheral **Note:** The DMAC sets davalid HIGH when it starts to provide valid control information on datype. The state of davalid and datype remain constant until the peripheral sets daready HIGH.
datype[1:0]	Indicates the type of acknowledgment, or request, that the DMAC signals: • b00 = The DMAC has completed the single DMA transfer • b01 = The DMAC has completed the burst DMA transfer • b10 = DMAC requesting the peripheral to perform a flush request • b11 = reserved

For more information, refer to the "Peripheral Request Interface Timing Diagrams" chapter.

Related Information

Peripheral Request Interface Timing Diagrams on page 16-20

Peripheral Request Interface Mapping

You can assign a peripheral request interface to any of the DMA channels. When a DMA channel thread executes DMAWFP, the value programmed in the peripheral [4:0] field specifies the peripheral associated with that DMA channel.

The DMAC supports 32 peripheral request handshakes. Each request handshake can receive up to four outstanding requests, and is assigned a specific peripheral device ID. The following table lists the peripheral device ID assignments.

Table 16-3: Peripheral Request Interface Mapping

Peripheral	Request Interface ID	Protocol
FPGA 0	0	Synopsys
FPGA 1	1	Synopsys
FPGA 2	2	Synopsys
FPGA 3	3	Synopsys
FPGA 4	4	Synopsys
FPGA 5/Security Manager DMA Tx	5	Synopsys
FPGA 6/I^2C EMAC2 Tx	6	Synopsys
FPGA 7/I^2C EMAC2 Rx	7	Synopsys
I^2C0 Tx	8	Synopsys
I^2C0 Rx	9	Synopsys
I^2C1 Tx	10	Synopsys
I^2C1 Rx	11	Synopsys
I^2C EMAC0 Tx	12	Synopsys
I^2C EMAC0 Rx	13	Synopsys
I^2C EMAC1 Tx	14	Synopsys
I^2C EMAC1 Rx	15	Synopsys
SPI0 Master Tx	16	Synopsys
SPI0 Master Rx	17	Synopsys
SPI0 Slave Tx	18	Synopsys
SPI0 Slave Rx	19	Synopsys
SPI1 Master Tx	20	Synopsys
SPI1 Master Rx	21	Synopsys

Peripheral	Request Interface ID	Protocol
SPI1 Slave Tx	22	Synopsys
SPI1 Slave Rx	23	Synopsys
Quad SPI Flash Tx	24	ARM
Quad SPI Flash Rx	25	ARM
STM	26	ARM
FPGA Manager DMA Rx	27	Synopsys
UART0 Tx	28	Synopsys
UART0 Rx	29	Synopsys
UART1 Tx	30	Synopsys
UART1 Rx	31	Synopsys

Mapping to a DMA Channel

The DMAC enables you to assign a peripheral request interface to any of the DMA channels. When a DMA channel thread executes `DMAWFP`, the value programmed in the peripheral [4:0] field specifies the peripheral associated with that DMA channel.

Related Information

DMAWFP on page 16-51

Request Acceptance Capability

During configuration of the DMAC, you can set the number of simultaneous active requests that a DMAC is able to accept, for each peripheral request interface. An active request is where the DMAC has not started the requested AXI data transfers.

The DMAC has a request FIFO, for each peripheral interface, which it uses to capture the requests from a peripheral. The depth of a FIFO depends on the number of simultaneous active requests that the corresponding peripheral request interface is configured to support. To store the state of an active request from the peripheral, the request FIFO uses two bits to store the state of:

- `drtype_<x>[0]` - Indicates the request type, burst or single
- `drlast_<x>` - Indicates if the peripheral is requesting the last data transfer of the DMA transfer

When a request FIFO is full, then the DMAC sets the corresponding `drready_<x>` to LOW to signal that the DMAC cannot accept any requests sent from the peripheral.

Peripheral Length Management

A peripheral can control the quantity of data that a DMA cycle contains, without the DMAC being aware of how many data transfers it contains. The peripheral controls the DMA cycle in one of the following ways:

- Selects a single transfer
- Selects a burst transfer
- Notifies the DMAC when it commences the final request in the current series

When the DMAC executes a `DMAWFP` peripheral instruction, it halts execution of the thread and waits for the peripheral to send a request. When the peripheral sends the request, the DMAC sets the request flags depending on the state of the following signals:

- `drtype_<x>[1:0]` —The DMAC sets the state of the `request_type` flag:
  - `drtype_<x>[1:0]`=b00—`request_type` <x> = Single
  - `drtype_<x>[1:0]`=b01—`request_type` <x> = Burst
- `drlast_<x>`—The DMAC sets the state of the `request_last` flag:
  - `drlast_<x>`=0—`request_last` <x> = 0
  - `drlast_<x>`=1—`request_last` <x> = 1

Note: If the DMAC executes a `DMAWFP` single or `DMAWFP` burst instruction then the DMAC sets:

- The request_type<x> flag to Single or Burst, respectively
- The request_last<x> flag to 0

`DMALPFE` is an assembler directive that forces the associated `DMALPEND` instruction to have its `nf` bit cleared, creating a program loop that does not use a loop counter to terminate the loop.

The DMAC exits the loop when the `request_last` flag is set.

The DMAC conditionally executes the following instructions, depending on the state of the `request_type` and `request_last` flags:

- **`DMALD`** , **`DMAST`** , **`DMALPEND`** - When these instructions use the optional B|S suffix then the DMAC executes a `DMANOP` if the `request_type` flag does not match.
- **`DMALDP<B|S>`** , **`DMASTP<B|S>`** - The DMAC executes a `DMANOP` if the `request_type` <x> flag does not match the B|S.
- **`DMALPEND`** - When the `nf` bit is 0, the DMAC executes a `DMANOP` if the `request_last` flag is set.

Use the `DMALDB`, `DMALDPB`, `DMASTB` and `DMASTPB` instructions if you require the DMAC to issue a burst transfer when the DMAC receives a burst request. The values in the `CCR`*n* register control the amount of data that the DMAC transfers.

Use the `DMALDS`, `DMALDPS`, `DMASTS` and `DMASTPS` instructions if you require the DMAC to issue a single transfer when the DMAC receives a single request. The DMAC ignores the value of the `src_burst_len` and `dst_burst_len` fields in the `CCR`*n* register and sets the `arlen[3:0]` or `awlen[3:0]` buses to 0x0.

Example Program for Peripheral Length Management

This example shows a DMAC program that transfers 64 words from memory to peripheral zero, when the peripheral sends a burst request, that is, `drtype_<x>[1:0] = b01`. When the peripheral sends a single request, that is, `drtype_<x>[1:0] = b00`, then the DMAC program transfers one word from memory to peripheral zero.

To transfer the 64 words, the program instructs the DMAC to perform 16 AXI transfers. Each AXI transfer consists of a 4-beat burst (SB=4, DB=4), each beat of which moves a word of data (SS=32, DS=32).

The following program shows the use of the:

- `DMAWFP periph` instruction - The DMAC waits for either a burst or single request from the peripheral.
- `DMASTPB and DMASTPS` instructions - The DMAC informs the peripheral when a transfer is complete.

Example 16-1: Peripheral Length Management Program

```
# Set up for burst transfers (4-beat burst, so SB4 and DB4), (word data
width, so SS32 and DS32)
DMAMOV CCR SB4 SS32 DB4 DS32
DMAMOV SAR ...
DMAMOV DAR ...
# Initialize peripheral '0'
DMAFLUSHP P0
# Perform peripheral transfers
# Outer loop - DMAC responds to peripheral requests until peripheral sets
drlast_0 = 1
DMALPFE
# Wait for request, DMAC sets request_type0 flag depending on the request
type it receives
DMAWFP 0, periph
# Set up loop for burst request: first 15 of 16 sets of transactions
# Note: B suffix - conditionally executed only if request_type0 flag = Burst
DMALP 15
DMALDB
DMASTB
# Only loop back if servicing a burst, otherwise treat as a NOP
DMALPENDB
# Perform final transaction (16 of 16). Send the peripheral acknowledgement
of burst request completion
DMALDB
DMASTPB P0
# Perform transaction if the peripheral signals a single request
# Note: S suffix - conditionally executed only if request_type0 flag = Single
DMALDS
DMASTPS P0
# Exit loop if DMAC receives the last request, that is, drlast_0 = 1
DMALPEND
DMAEND
```

DMAC Length Management

The DMAC controls the total amount of data. The peripheral uses the peripheral request interface to notify the DMAC when it requires the DMAC to transfer data to or from the peripheral. The DMA channel thread controls how the DMAC responds to the peripheral requests.

The following constraints apply to DMAC length management:

- The total quantity of data for all the single requests from a peripheral must be less than the quantity of data for a burst request for that peripheral.

  Note: The `CCRn` register controls how much data is transferred for a burst request and a single request. Altera recommends that you do not update a `CCRn` register when a transfer is in progress for channel `n`.

- After the peripheral sends a burst request, the peripheral must not send a single request until the DMAC acknowledges that the burst request is complete.

Program the DMAWFP single instruction when you require the program thread to halt execution until the peripheral request interface receives any request type.

- Single— If the head entry in the request FIFO buffer is a single request type, the DMAC pops the entry from the FIFO buffer and continues program execution.
- Burst—If the head entry in the request FIFO buffer is a burst request type, the DMAC leaves the entry in the FIFO buffer and continues program execution.

Note: The burst request entry remains in the request FIFO buffer until the DMAC executes a DMAWFP burst instruction or a DMAFLUSHP instruction.

Program the DMAWFP burst instruction when you require the program thread to halt execution until the peripheral request interface receives a burst request.

- Single—If the head entry in the request FIFO buffer is a single request type, the DMAC removes the entry from the FIFO buffer and continues program execution.
- Burst—If the head entry in the request FIFO buffer is a burst request type, the DMAC pops the entry from the FIFO buffer and continues program execution.

Program the DMALDP instruction when you require the DMAC to send an acknowledgment to the peripheral when it completes the AXI read transfers. Similarly, use the DMASTP instruction when you require the DMAC to send an acknowledgment to the peripheral when it completes the AXI write transfers. The DMAC uses the acknowledge bus to signal a transfer acknowledgment to the peripheral.

Note: The DMAC sends an acknowledgment for a read transaction when the rvalid and rlast signals are high and for a write transaction when bvalid signal is high. The DMAC might send an acknowledgment to the peripheral while the transfer of write data to the end destination is still in progress.

Use the DMAFLUSHP instruction to reset the request FIFO buffer for the peripheral request interface. After the DMAC executes DMAFLUSHP, it ignores peripheral requests until the peripheral acknowledges the flush request. This protocol enables the DMAC and peripheral to synchronize with each other.

Example Program for DMAC Length Management

This example shows a DMAC program that can transfer 1027 words when a peripheral signals 16 consecutive burst requests and three consecutive single requests.

Example 16-2: DMAC Length Management Program

```
# Set up for AXI burst transfer (4-beat burst, so SB4 and DB4), (word data
width, so SS32 and DS32)
DMAMOV CCR SB4 SS32 DB4 DS32
DMAMOV SAR ...
DMAMOV DAR ...
# Initialize peripheral '0'
DMAFLUSHP P0
# Perform peripheral transfers
# Burst request loop to transfer 1024 words
DMALP 16
# Wait for the peripheral to signal a burst request. DMAC transfers 64 words
for each burst request
DMAWFP 0, burst
# Set up loop for burst request: first 15 of 16 sets of transactions
DMALP 15
DMALD
DMAST
```

```
DMALPEND
# Perform final transaction (16 of 16). Send the peripheral acknowledgement
of burst request completion
DMALD
DMASTPB 0
# Finish burst loop
DMALPEND
# Set up for AXI single transfer (word data width, so SS32 and DS32)
DMAMOV CCR SB1 SS32 DB1 DS32
# Single request loop to transfer 3 words
DMALP 3
# Wait for the peripheral to signal a single request. DMAC to transfer one
word
DMAWFP 0, single
# Perform transaction for single request and send completion acknowledgement
to the peripheral
DMALDS
DMASTPS P0
# Finish single loop
DMALPEND
# Flush the peripheral, in case the single transfers were in response to a
burst request
DMAFLUSHP 0
DMAEND
```

Peripheral Request Interface Timing Diagrams

The following are examples of the functional operation of the peripheral request interface using the rules that the "Handshake Rules" chapter describes.

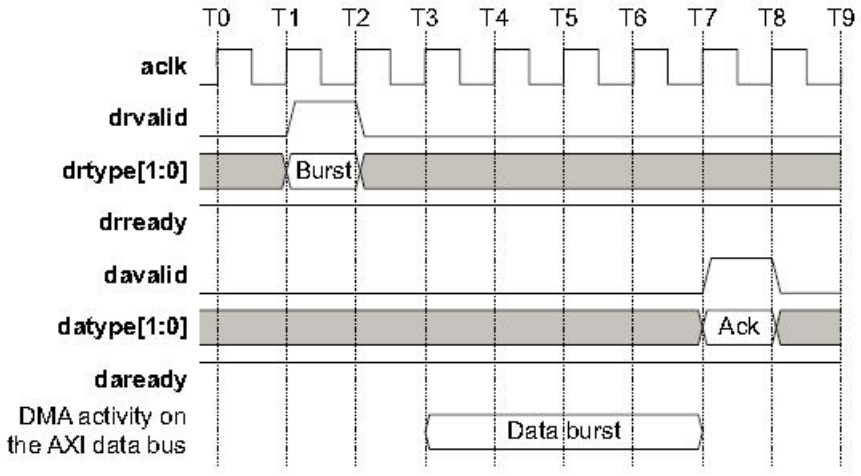
Related Information

Handshake Rules on page 16-13

Burst Request

DMA request timing when a peripheral requests a burst transfer.

Figure 16-4: Burst Request Signaling

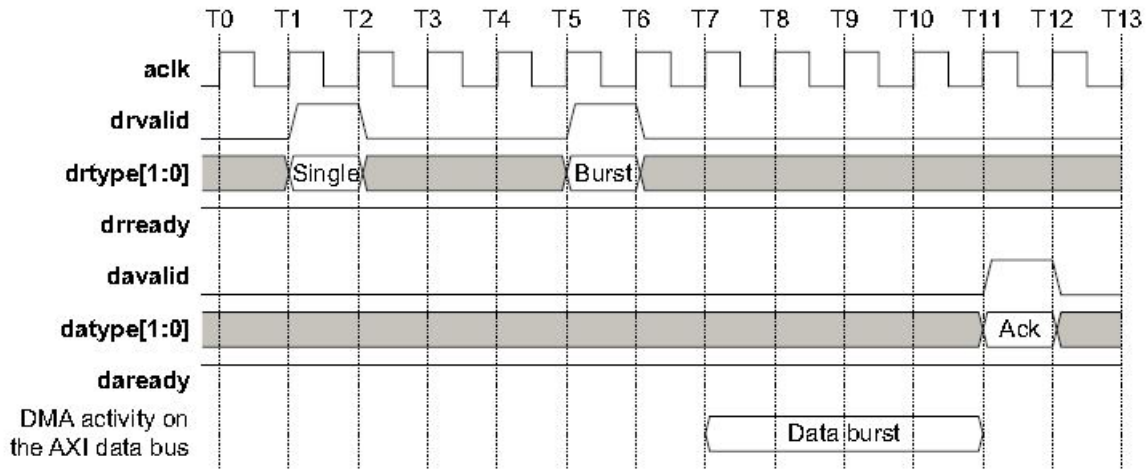


Timing	Description
T1	The DMAC detects a request for a burst transfer.
T3 - T6	The DMAC performs a burst transfer.
T7	The DMAC sets `davalid` HIGH and sets `datype[1:0]` to indicate that the burst transfer is complete.

Single and Burst Request

DMA request timing when a peripheral requests a single and a burst transfer.

Figure 16-5: Single and Burst Request Signaling



Timing	Description
T1	The DMAC detects a request for a single transfer.
T3	The DMAC ignores the single transfer request because the DMA channel thread had executed a DMAWFP burst instruction.
T5	The DMAC detects a request for a burst transfer.
T7 - T10	The DMAC performs a burst transfer.
T11	The DMAC sets `davalid` HIGH and sets `datype[1:0]` to indicate that the burst transfer is complete.

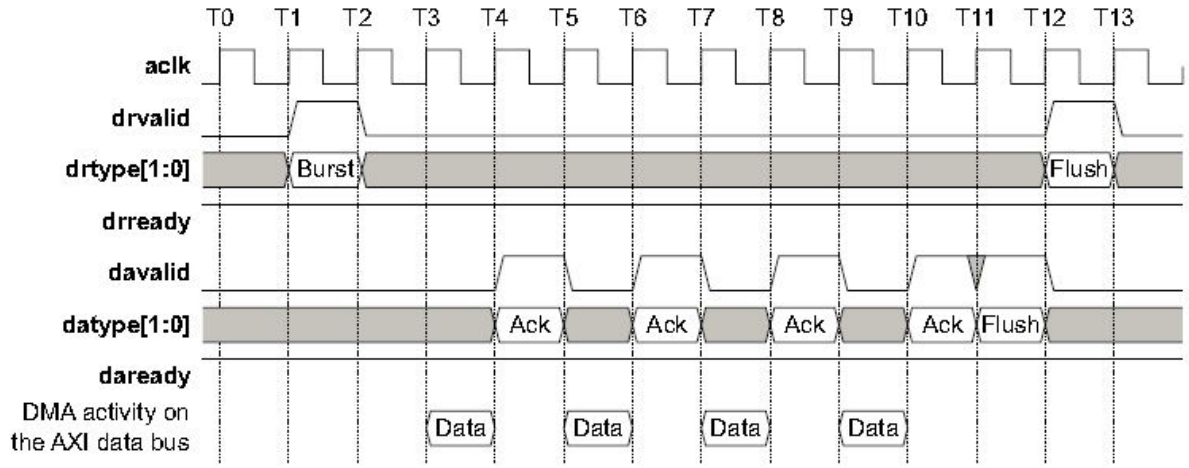
Related Information

DMAWFP on page 16-51

DMAC Performs Single Transfers for a Burst Request

DMA request timing when a peripheral requests a burst transfer, but the DMAC has insufficient data remaining in the MFIFO to generate a burst and therefore completes the request using single transfers.

Figure 16-6: Single Transfers for a Burst Request



Timing	Description
T1	The DMAC detects a request for a burst transfer.
T3	The MFIFO contains insufficient data for the DMAC to generate a burst transfer and therefore, the DMAC performs a single transfer.
T4	The DMAC signals `davalid` and `datype[1:0]` to indicate completion of a single transfer.
T5 - T10	The DMAC performs the remaining three single transfers.
T11	The DMAC signals `davalid` and `datype[1:0]` to request the peripheral to flush the contents of any control registers that are associated with the current DMA cycle.
T12	The peripheral signals `drvalid` and `drtype[1:0]` to acknowledge the flush request.

Using Events and Interrupts

The DMAC can support up to eight events and interrupts. The `INTEN` register determines whether each event-interrupt resource is an event or an interrupt.

When the DMAC executes a `DMASEV` instruction, it modifies the event-interrupt resource that you specify. The `INTEN` register sets the event-interrupt resource to function as an:

- **Event**—The DMAC generates an event for the specified event-interrupt resource. When the DMAC executes a `DMAWFE` instruction for the same event-interrupt resource then it clears the event.
- **Interrupt**—The DMAC sets the `irq` <event_num> signal high, where <event_num> is the number of the specified event resource. To clear the interrupt you must write to the `INTCLR` register.

Using an Event to Restart DMA Channels

When you program the `INTEN` register to generate an event, you can use the `DMASEV` and `DMAWFE` instructions to restart one or more DMA channels.

DMAC executes DMAWFE before DMASEV

To restart a single DMA channel:

1. The first DMA channel executes `DMAWFE` and then stalls while it waits for the event to occur.
2. The other DMA channel executes `DMASEV` using the same event number generating an event and restarting the first DMA channel. The DMAC clears the event, one clock cycle after it executes `DMASEV`.

You can program multiple channels to wait for the same event. For example, if four DMA channels have all executed `DMAWFE` for event 2, and another DMA channel executes `DMASEV` for event 2, the four DMA channels all restart at the same time. The DMAC clears the event, one clock cycle after it executes `DMASEV`.

DMAC executes DMASEV before DMAWFE

If the DMAC executes `DMASEV` before another channel executes `DMAWFE`, the event remains pending until the DMAC executes `DMAWFE`. When the DMAC executes `DMAWFE`, it halts execution for one `aclk` clock cycle, clears the event and then continues execution of the channel thread.

For example, if the DMAC executes `DMASEV 6` and none of the other threads have executed `DMAWFE 6`, then the event remains pending. If the DMAC executes the `DMAWFE 6` instruction for channel 4 and then executes the `DMAWFE 6` instruction for channel 3, the following actions occur:

1. The DMAC halts execution of the channel 4 thread for one clock cycle.
2. The DMAC clears event 6.
3. The DMAC resumes execution of the channel 4 thread.
4. The DMAC halts execution of the channel 3 thread and the thread stalls while it waits for the next occurrence of event 6.

Interrupting the MPU Subsystem

The DMAC provides the `irq[x]` signals for use as active-high level-sensitive interrupts to the MPU subsystem. When you program the `INTEN` register to generate an interrupt, after the DMAC executes `DMASEV`, it sets the corresponding `irq[x]` signal high.

The MPU subsystem can clear the interrupt by writing to the `INTCLR` register.

Note: Executing `DMAWFE` does not clear an interrupt.

If you use the `DMASEV` instruction to notify a microprocessor when the DMAC completes a `DMALD` or `DMAST` instruction then you should insert a memory barrier instruction before the `DMASEV`. Otherwise the DMAC might signal an interrupt before the AXI transfers complete.

The following program shows the example of Memory Barrier Instruction.

```
DMALD
DMAST
# Issue a write memory barrier
# Wait for the AXI write transfer to complete before the DMAC
# can send an interrupt
DMAWMB
# The DMAC sends the interrupt
DMASEV
```

Aborts

Abort Types

An abort can be classified as either precise or imprecise, depending on whether the DMAC provides an abort handler with the precise state of the DMAC when the abort occurs.

- Precise Abort - The DMAC updates the `PC` register with the address of the instruction that created the abort.
- Imprecise Abort - The `PC` register might contain the address of an instruction that did not cause the abort to occur.

Abort Sources

The DMAC indicates a precise abort under any of the following conditions:

- A DMA channel thread in the Non-secure state attempts to program its `CCRn` register and generate a secure AXI transaction.
- A DMA channel thread in the Non-secure state executes `DMAWFE` or `DMASEV` for an event that is set as secure. The `boot_irq_ns` memory-mapped control signals initialize the security state for an event.

Note: For each event, the `INTEN` register controls if the DMAC generates an event or signals an interrupt.

- A DMA channel thread attempts to execute `DMAST` but the DMAC calculates that when it eventually performs the store, the MFIFO buffer contains insufficient data to enable it to complete the store.
- A DMA channel thread in the Non-secure state executes `DMAWFP`, `DMALDP`, `DMASTP`, or `DMAFLUSHP` for a peripheral request interface that is set as secure. The `boot_periph_ns` memory-mapped control signals initialize the security state for a peripheral request interface.
- A DMA manager thread in the Non-secure state executes `DMAGO` to attempt to start a secure DMA channel thread.
- The DMAC receives an ERROR response on the AXI master interface when it performs an instruction fetch.
- A thread executes an undefined instruction.
- A thread executes an instruction with an operand that is invalid for the configuration of the DMAC.

Note: When the DMAC signals a precise abort, the instruction that triggers the abort is not executed. Instead, the DMAC executes a `DMANOP`.

The DMAC signals an imprecise abort under the following conditions:

- The DMAC receives an ERROR response on the AXI master interface when it performs a data load.
- The DMAC receives an ERROR response on the AXI master interface when it performs a data store.
- A DMA channel thread executes `DMALD` or `DMAST`, and the MFIFO buffer is too small to hold the required amount of data.
- A DMA channel thread executes `DMAST` but the thread has not executed sufficient `DMALD` instructions.
- A DMA channel thread locks up because of resource starvation, and this causes the internal watchdog timer to time out.

Related Information

ARM Information Center

For more information about ARM's DMA-330 controller, refer to the *CoreLink DMA Controller DMA-330 Revision: r1p2* Technical Reference Manual on the ARM Infocenter website.

Watchdog Abort

The DMAC can lock up if one or more DMA channel programs are running and the MFIFO buffer is too small to satisfy the storage requirements of the DMA programs.

The DMAC contains logic to prevent it from remaining in a state where it is unable to complete a DMA transfer.

The DMAC detects a lock up when all of the following conditions occur:

- Load queue is empty.
- Store queue is empty.
- All of the running channels are prevented from executing a `DMALD` instruction either because the MFIFO buffer does not have sufficient free space or another channel owns the load-lock.

When the DMAC detects a lockup, it signals an interrupt and can also abort the contributing channels. The DMAC behavior depends on the state of the `wd_irq_only` bit in the `WD` register, if:

- `wd_irq_only=0`—The DMAC aborts all of the contributing DMA channels and sets the `irq_abort` signal high.
- `wd_irq_only=1`—The DMAC sets the `irq_abort` signal high.

Related Information

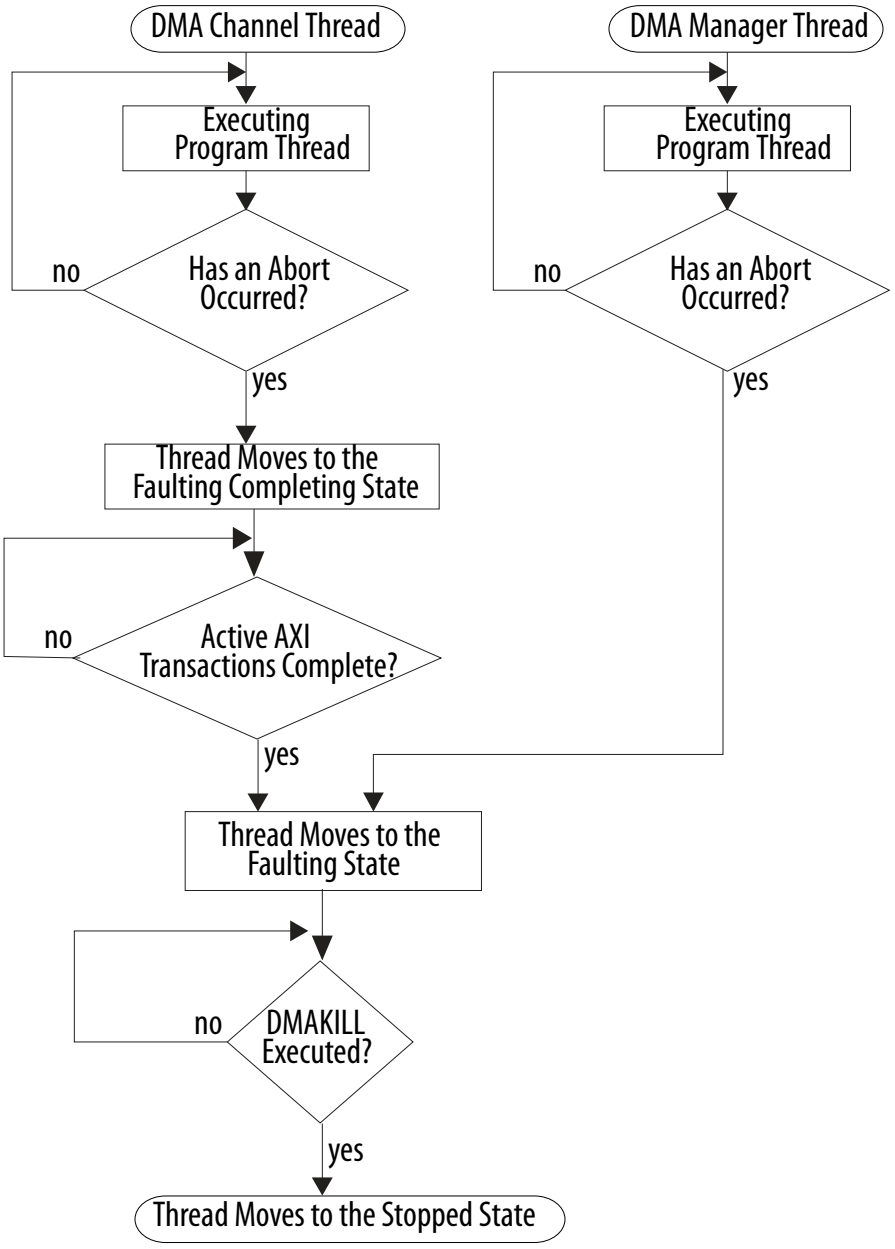
Resource Sharing Between DMA Channels on page 16-32

Abort Handling

The architecture of the DMAC is not designed to recover from an abort. You must use an external agent, such as the MPU, to terminate a thread when an abort occurs.

The following figure shows the operating states for the DMA channel and DMA manager threads after an abort occurs.

Figure 16-7: Abort Process



After an abort occurs, the action of the DMAC depends on the thread type:

- DMA channel thread—The thread immediately moves to the Faulting completing state. In this state, the DMAC performs the following operations:
  - Sets the `irq_abort` signal high.
  - Stops executing instructions for the DMA channel.
  - Invalidates all cache entries for the DMA channel updates the `CPC`*n* register to contain the address of the aborted instruction provided that the abort is precise.
  - Does not generate AXI accesses for any instructions remaining in the read queue and write queue.
  - Permits currently active AXI transactions to complete.

  Note: After the transactions for the DMA channel finish, the thread moves to the Faulting state.
- DMA manager thread—The thread immediately moves to the Faulting state and the DMAC sets the `irq_abort` signal high.

  The external agent can respond to the assertion of the `irq_abort` signal by all of the following:

  - Reading the status of the `FSRD` register to determine if the DMA manager is Faulting. In the Faulting state, the `FSRD` register provides the cause of the abort.
  - Reading the status of the `FSRC` register to determine if a DMA channel is Faulting. In the Faulting state, the `FSRC` register provides the cause of the abort.

  To enable a thread in the Faulting state to move to the Stopped state, the external agent must:

  - Program the `DBGINST0` register with the encoding for the `DMAKILL` instruction.
  - Write to the `DBGCMD` register.

  Note: If the aborted thread is secure, you must use the secure slave interface to update these registers.

  After a thread in the Faulting state executes `DMAKILL`, it moves to the Stopped state.

Security Usage

When the DMAC exits from reset, the status of the configuration signals configures the security for:

- DMA manager thread—The `DNS` bit in the DSR register returns the security state of the DMA manager thread.
- Events and interrupts—The `INS` bit in the `CR3` register returns the security state of the event-interrupt resources.
- Peripheral request interfaces—The `PNS` bit in the `CR4` register returns the security state of these interfaces.

Additionally, each DMA channel thread contains a dynamic non-secure bit, `CNS`, that is valid when the channel is not in the Stopped state.

DMA Manager Thread in Secure State

If the DNS bit is 0, the DMA manager thread operates in the secure state and performs only secure instruction fetches. When a DMA manager thread in the secure state processes:

- DMAGO—The DMAC uses the status of the ns bit, to set the security state of the DMA channel thread by writing to the CNS bit for that channel.
- DMAWFE—The DMAC halts execution of the thread until the event occurs. When the event occurs, the DMAC continues execution of the thread, irrespective of the security state of the corresponding INS bit.
- DMASEV—The DMAC sets the corresponding bit in the INT_EVENT_RIS register, irrespective of the security state of the corresponding INS bit.

DMA Manager Thread in Non-Secure State

If the DNS bit is 1, the DMA manager thread operates in the non-secure state, and it only performs non-secure instruction fetches. When a DMA manager thread in the non-secure state processes:

DMAGO - The DMAC uses the status of the ns bit to control if it starts a DMA channel thread.

- If ns = 0, then the DMAC does not start a DMA channel thread and instead it:
  - Executes an NOP
  - Sets the FSRD register
  - Sets the dmago_err bit in the FTRD register
  - Moves the DMA manager to the Faulting state
- If ns = 1, then the DMAC starts a DMA channel thread in the non-secure state and programs the CNS bit to be non-secure.

DMAWFE - The DMAC uses the status of the corresponding INS bit in the CR3 register to control whether it waits for the event.

- If INS = 0, then the event is in the secure state. The DMAC:
  - Executes an NOP
  - Sets the FSRD register
  - Sets the mgr_evnt_err bit in the FTRD register
  - Moves the DMA manager to the Faulting state
- If INS = 1, then the event is in the non-secure state. The DMAC halts execution of the thread and waits for the event to occur.

DMASEV - The DMAC uses the status of the corresponding INS bit in the CR3 register to control if it creates the event interrupt.

- If INS = 0, then the event-interrupt resource is in the secure state. The DMAC:
  - Executes a NOP
  - Sets the FSRD register
  - Sets the mgr_evnt_err bit in the FTRD register
  - Moves the DMA manager to the Faulting state
- If INS = 1, then the event-interrupt resource is in the non-secure state. The DMAC creates the event interrupt.

DMA Channel Thread in Secure State

When the CNS bit is 0, the DMA channel thread is programmed to operate in the secure state and to only perform secure instruction fetches.

When a DMA channel thread in the secure state processes the following instructions:

- DMAWFE - The DMAC halts execution of the thread until the event occurs. When the event occurs, the DMAC continues execution of the thread, regardless of the security state of the corresponding INS bit in the CR3 register.
- DMASEV - The DMAC creates the event interrupt, regardless of the security state of the corresponding INS bit in the CR3 register.
- DMAWFP - The DMAC halts execution of the thread until the peripheral signals a DMA request at which point the DMAC continues execution of the thread, regardless of the security state of the corresponding PNS bit in the CR4 register.
- DMALDP and DMASTP - The DMAC sends a message to the peripheral to communicate that data transfer is complete, regardless of the security state of the corresponding PNS bit in the CR4 register.
- DMAFLUSHP - The DMAC clears the state of the peripheral and sends a message to the peripheral to resend its level status, regardless of the security state of the corresponding PNS bit in the CR4 register.

When a DMA channel thread is in the secure state, it enables the DMAC to perform secure and non-secure AXI accesses.

DMA Channel Thread in Non-Secure State

When the CNS bit is 1, the DMA channel thread is programmed to operate in the non-secure state and to only perform non-secure instruction fetches.

When a DMA channel thread in the non-secure state processes the following instructions:

DMAWFE - The DMAC uses the status of the corresponding INS bit in the CR3 register, to control if it waits for the event.

- If INS =0, then the event is in the secure state. The DMAC:
  - Executes an NOP
  - Sets the appropriate bit in the FSRC register corresponding to the DMA channel number
  - Sets the ch_evnt_err bit in the FTRn register
  - Moves the DMA channel to the Faulting completing state
- If INS =1, then the event is in the non-secure state. The DMAC halts execution of the thread and waits for the event to occur.

DMASEV - The DMAC uses the status of the corresponding INS bit in the CR3 register, to control if it creates the event.

- If INS =0, then the event-interrupt resource is in the secure state. The DMAC:
  - Executes an NOP
  - Sets the appropriate bit in the FSRC register corresponding to the DMA channel number
  - Sets the ch_evnt_err bit in the FTRn register
  - Moves the DMA channel to the Faulting completing state
- If INS =1, then the event-interrupt resource is in the non-secure state. The DMAC creates the event interrupt.

DMAWFP - The DMAC uses the status of the corresponding PNS bit in the CR4 register, to control if it waits for the peripheral to signal a request.

- If PNS =0, then the peripheral is in the secure state. The DMAC:
  - Executes an NOP
  - Sets the appropriate bit in the FSRC register corresponding to the DMA channel number
  - Sets the ch_periph_err bit in the FTRn register
  - Moves the DMA channel to the Faulting completing state
- If PNS =1, then the peripheral is in the non-secure state. The DMAC halts execution of the thread and waits for the peripheral to signal a request.

DMALDP and DMASTP - The DMAC uses the status of the corresponding PNS bit in the CR4 register, to control if it sends an acknowledgement to the peripheral.

- If PNS =0, then the peripheral is in the secure state. The DMAC:
  - Executes an NOP
  - Sets the appropriate bit in the FSRC register corresponding to the DMA channel number
  - Sets the ch_periph_err bit in the FTRn register
  - Moves the DMA channel to the Faulting completing state
- If PNS =1, then the peripheral is in the non-secure state. The DMAC sends a message to the peripheral to communicate when the data transfer is complete.

DMAFLUSHP - The DMAC uses the status of the corresponding PNS bit in the CR4 register, to control if it sends a flush request to the peripheral.

- If PNS =0, then the peripheral is in the secure state. The DMAC:
  - Executes an NOP
  - Sets the appropriate bit in the FSRC register corresponding to the DMA channel number
  - Sets the ch_periph_err bit in the FTRn register
  - Moves the DMA channel to the Faulting completing state
- If PNS =1, then the peripheral is in the non-secure state. The DMAC clears the state of the peripheral and sends a message to the peripheral to resend its level status.

When a DMA channel thread is in the non-secure state, and a DMAMOV CCR instruction attempts to program the channel to perform a secure AXI transaction, the DMAC:

1. Executes a DMANOP
2. Sets the appropriate bit in the FSRC register corresponding to the DMA channel number
3. Sets the ch_rdwr_err bit in the FTR*n* register
4. Moves the DMA channel thread to the Faulting completing state

Programming Restrictions

Certain restrictions apply when programming the DMAC.

Fixed Unaligned Bursts

The DMAC does not support fixed unaligned bursts. The DMAC treats the following conditions as programming errors:

- **Unaligned read**
  - src_inc field is 0 in the CCRn register.
  - The SARn register contains an address that is not aligned to the size of data that the src_burst_size field contains.
- **Unaligned write**
  - dst_inc field is 0 in the CCRn register.
  - The DARn register contains an address that is not aligned to the size of data that the dst_burst_size field contains.

Endian Swap Size Restrictions

If you program the endian_swap_size field in the CCRn register, to enable a DMA channel to perform an endian swap, then you must set the corresponding SAR register and the corresponding DARn register to contain an address that is aligned to the size that the endian_swap_size field specifies before executing any DMALD or DMAST instructions.

Note: If you update any of endian_swap_size, SARn, or DARn, for example, using a DMAADDH SAR instruction, then you must ensure that the SARn and DARn registers contain an address aligned to the size that the endian_swap_size field specifies before executing any additional DMALD or DMAST instructions.

If you program the src_inc field in the CCR register to use a fixed address, you must program the src_burst_size field to select a burst size that is greater than or equal to the value that the endian_swap_size field specifies. Similarly, if you program the dst_inc field to select a fixed destination address, you must program the dst_burst_size field to select a burst size that is greater than or equal to the value that the endian_swap_size field specifies.

If you program the dst_inc field in the CCRn register to use an incrementing address, you must program the CCRn register so that dst_burst_len times dst_burst_size is a multiple of endian_swap_size. For example, if endian_swap_size = 2, 32-bit, and dst_burst_size = 1, 2 bytes per beat, then you can program dst_burst_len = 1, 3, 5, ..., 15, that is 2, 4, 6, ..., 16 transfers.

Updating DMA Channel Control Registers During a DMA Cycle

Prior to the DMAC executing a sequence of DMALD and DMAST instructions, the values you program in to the CCRn register, SARn register, and DARn register control the data byte lane manipulation that the DMAC performs when it transfers the data from the source address to the destination address.

You can update these registers during a DMA cycle, but if you change certain register fields, then the DMAC may discard data. The following sections describe the register fields that could have a detrimental impact on a data transfer.

Updates that affect the destination address

If you use a DMAMOV instruction to update the DARn register or CCRn register part way through a DMA cycle, then there may be a discontinuity in the destination data stream.

A discontinuity occurs if you change any of the following:

- `endian_swap_size` field.
- `dst_inc` bit.
- `dst_burst_size` field when `dst_inc` = 0, that is, fixed-address burst.
- `DARn` register so that it modifies the destination byte lane alignment. Because the bus width is 64 bits, you change bits [2:0] in the `DARn` register.

When a discontinuity in the destination data stream occurs, the DMAC:

1. Halts execution of the DMA channel thread.
2. Completes all outstanding read and write operations for the channel. That is, as if the DMAC were executing `DMARMB` and `DMAWMB`.
3. Discards any residual MFIFO buffer data for the channel.
4. Resumes execution of the DMA channel thread.

Updates that affect the source address

If you use a `DMAMOV` instruction to update the `SARn` register or `CCRn` register part way through a DMA cycle, then this might cause a discontinuity in the source data stream.

A discontinuity occurs if you change any of the following:

- `src_inc` bit.
- `src_burst_size` field.
- `SAR` register so that it modifies the source byte lane alignment. Because the bus width is 64 bits, you change bits [2:0] in the `SAR` register.

When a discontinuity in the source data stream occurs, the DMAC:

1. Halts execution of the DMA channel thread.
2. Completes all outstanding read operations for the channel. That is, as if the DMAC were executing `DMARMB`
3. Resumes execution of the DMA channel thread. No data is discarded from the MFIFO buffer.

Resource Sharing Between DMA Channels

DMA channel programs share the MFIFO buffer data storage resource. You must not start a set of concurrently running DMA channel programs with a resource requirement that exceeds 512, the size of the MFIFO buffer. If you exceed this limit, then the DMAC might lock up and generate a Watchdog abort.

Refer to the "Watchdog Abort" section for more information regarding the abort mechanism.

The DMAC includes a mechanism called the *load-lock* to ensure that the shared MFIFO buffer resource is used correctly. The load-lock is either owned by one channel or it is free. The channel that owns the load-lock can execute `DMALD` instructions successfully. A channel that does not own the load-lock pauses at a `DMALD` instruction until it takes ownership of the load-lock.

A channel claims ownership of the load-lock when:

- It executes a `DMALD` or `DMALDP` instruction
- No other channel currently owns the load-lock

A channel releases ownership of the load-lock when any of the following occur:

- It executes a `DMAST`, `DMASTP`, or `DMASTZ`
- It reaches a barrier by executing `DMARMB` or `DMAWMB`
- It waits by executing `DMAWFP` or `DMAWFE`
- It terminates normally, that is, it executes `DMAEND`
- It aborts for any reason, including `DMAKILL`

The MFIFO buffer resource usage of a DMA channel program is measured in MFIFO buffer entries, and rises and falls as the program proceeds. The MFIFO buffer resource requirement of a DMA channel program is described using a *static requirement* and a *dynamic requirement* which are affected by the load-lock mechanism.

The static requirement is defined to be the maximum number of MFIFO buffer entries that a channel is currently using before that channel does one of the following:

- Executes a WFP or WFE instruction
- Claims ownership of the load-lock.

The dynamic requirement is defined to be the difference between the static requirement and the maximum number of MFIFO buffer entries that a channel program uses at any time during its

To calculate the total MFIFO buffer requirement, add the largest dynamic requirement to the sum of all the static requirements.

To avoid DMAC lockup, the total MFIFO buffer requirement of the set of channel programs must be equal to or less than 512, the MFIFO buffer depth.

Related Information

- **Watchdog Abort** on page 16-25
- **MFIFO Buffer Usage Overview** on page 16-54

Clocks and Resets

Clock

The DMA controller operates on the `l4_main_clk` input.

The DMA controller interfaces with the HPS peripherals through the peripheral request interface (PRI). The peripherals are in different clock domains, so clock crossing adapters are required between the DMA Controller and the peripherals.

Related Information

Clock Manager on page 2-1

Resets

The DMA controller has nine reset signals. The reset manager drives the `aresetn` signal to the DMA controller on a cold or warm reset. The reset manager drives the `dma_fpga_if_rst_n[7:0]` signals to reset the eight FPGA PRIs.

Table 16-4: Reset inputs to the DMA controller

Reset Signal	Description
`aresetn`	Resets DMA controller
`dma_fpga_if_rst_n[7:0]`	Resets the eight FPGA peripheral request interfaces

Related Information

Reset Manager on page 3-1

Taking the DMA Controller Out of Reset

When a cold or warm reset is issued in the HPS, the reset manager resets this module and holds it in reset until software releases it.

After the Cortex-A9 MPCore CPU boots, it can deassert the reset signal by clearing the appropriate bits in the reset manager's corresponding reset register. For details about reset registers, refer to "Module Reset Signals".

Related Information

- **Reset Manager** on page 3-1
- **Error Checking and Correction Controller** on page 11-1

Error Checking and Correction (ECC) controllers provide single- and double-bit error memory protection for integrated on-chip RAM and peripheral RAMs within the hard processor system (HPS).

Constraints and Limitations of Use

DMA Channel Arbitration

The DMAC uses a round-robin scheme to serve the active DMA channels. To ensure that the DMAC continues to serve the DMA manager, it always serves the DMA manager prior to serving the next DMA channel.

Note: You cannot alter the arbitration process of the DMAC.

DMA Channel Prioritization

The DMAC responds to all active DMA channels with equal priority. You cannot increase the priority of a DMA channel over any other DMA channels.

Instruction Cache Latency

When a cache miss occurs, most of the delay is introduced by the memory containing the DMA code; the DMAC adds minimal delay.

AXI Data Transfer Size

The DMAC can only perform data accesses up to 64 bits in width. If you program the `src_burst_size` or `dst_burst_size` fields to be larger, the DMAC indicates a precise abort.

Related Information

Abort Sources on page 16-24

AXI Bursts Crossing 4 KB Boundaries

The AXI specification does not permit AXI bursts to cross 4 KB address boundaries. If you program the DMAC with a combination of burst start address, size, and length that would cause a single burst to cross a 4 KB address boundary, then the DMAC generates a pair of bursts with a combined length equal to that specified. This operation is transparent to the DMAC channel thread program so that, for example, the DMAC responds to a single `DMALD` instruction by generating the appropriate pair of AXI read bursts.

AXI Burst Types

You can program the DMAC to generate only fixed-address or incrementing-address burst types for data accesses. It does not generate wrapping-address bursts for data accesses or for instruction fetches.

AXI Write Addresses

The DMAC can issue up to eight outstanding write addresses. The DMAC does not issue a write address until it has read in all of the data bytes required to fulfill that write transaction.

AXI Write Data Interleaving

The DMAC does not generate interleaved write data. All write data beats for one write transaction are output before any write data beat for the next write transaction.

DMA Controller Programming Model

Instruction Syntax Conventions

The following conventions are used in assembler syntax prototype lines and their subfields:

- `< >` - Any item bracketed by < and > is mandatory. A description of the item and of how it is encoded in the instruction is supplied by subsequent text.
- `[ ]` - Any item bracketed by [and] is optional. A description of the item and of how its presence or absence is encoded in the instruction is supplied by subsequent text.
- `" "` (spaces) - To separate items, single spaces are used for clarity. When a space is obligatory in the assembler syntax, two or more consecutive spaces are used.

Instruction Set Summary

The DMAC instructions:

- Indicate a DMA prefix, to provide a unique name-space
- Have 8-bit opcodes that might use a variable data payload of 0, 8, 16, or 32 bits
- Indicate suffixes that are consistent.

Table 16-5: Instruction Syntax Summary

Mnemonic	Instruction	DMA Manager Usage	DMA Channel Usage	Description
DMAADDH	Add Halfword	No	Yes	**DMAADDH** on page 16-37

Mnemonic	Instruction	DMA Manager Usage	DMA Channel Usage	Description
DMAADNH	Add Negative Halfword	No	Yes	**DMAADNH** on page 16-37
DMAEND	End	Yes	Yes	**DMAEND** on page 16-38
DMAFLUSHP	Flush and Notify Peripheral	No	Yes	**DMAFLUSHP** on page 16-38
DMAGO	Go	Yes	No	**DMAGO** on page 16-39
DMAKILL	Kill	Yes	Yes	**DMAKILL** on page 16-40
DMALD	Load	No	Yes	**DMALD[S | B]** on page 16-41
DMALDP	Load and Notify Peripheral	No	Yes	**DMALDP<S | B>** on page 16-42
DMALP	Loop	No	Yes	**DMALP** on page 16-43
DMALPEND	Loop End	No	Yes	**DMALPEND[S | B]** on page 16-43
DMALPFE	Loop Forever	No	Yes	**DMALPFE** on page 16-45
DMAMOV	Move	No	Yes	**DMAMOV** on page 16-45
DMANOP	No Operation	Yes	Yes	**DMANOP** on page 16-46
DMARMB	Read Memory Barrier	No	Yes	**DMARMB** on page 16-46
DMASEV	Send Event	Yes	Yes	**DMASEV** on page 16-47
DMAST	Store	No	Yes	**DMAST[S | B]** on page 16-48
DMASTP	Store and Notify Peripheral	No	Yes	**DMASTP<S | B>** on page 16-49
DMASTZ	Store Zero	No	Yes	**DMASTZ** on page 16-49
DMAWFE	Wait For Event	Yes	Yes	**DMAWFE** on page 16-50
DMAWFP	Wait For Peripheral	No	Yes	**DMAWFP** on page 16-51

Mnemonic	Instruction	DMA Manager Usage	DMA Channel Usage	Description
DMAWMB	Write Memory Barrier	No	Yes	**DMAWMB** on page 16-51

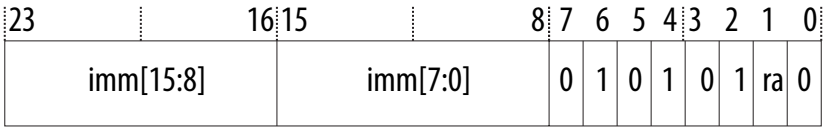
Instructions

DMAADDH

Add Halfword adds an immediate 16-bit value to the SARn register or DARn register, for the DMA channel thread. This enables the DMAC to support two-dimensional DMA operations.

Note: The immediate unsigned 16-bit value is zero-extended before the DMAC adds it to the address, using 32-bit addition. The DMAC discards the carry bit so that addresses wrap from 0xFFFFFFFF to 0x00000000.

Figure 16-8: DMAADDH Instruction Encoding



Assembler syntax

```
DMAADDH <address_register>, <16-bit bit immediate>
```

where:

`<address_register>` Selects the address register to use. It must be either:

- SAR SAR*n* register and sets `ra` to 0
- DAR DAR*n* register and sets `ra` to 1

`<16-bit immediate>` The immediate value to be added to the `<address_register>`.

Operation

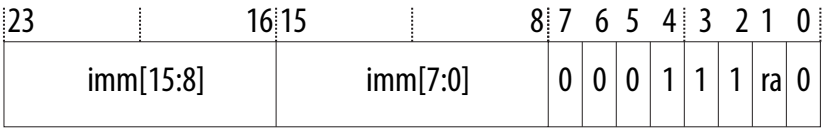
You can only use this instruction in a DMA channel thread.

DMAADNH

Add Negative Halfword adds an immediate negative 16-bit value to the SARn register or DARn register, for the DMA channel thread. This enables the DMAC to support two-dimensional DMA operations, or reading or writing an area of memory in a different order to naturally incrementing addresses.

Note: The immediate unsigned 16-bit value is one-extended to 32 bits, to create a value that is the two's complement representation of a negative number between -65536 and -1, before the DMAC adds it to the address using 32-bit addition. The DMAC discards the carry bit so that addresses wrap from 0xFFFFFFFF to 0x00000000. The net effect is to subtract between 65536 and 1 from the current value in the source or destination address register.

Figure 16-9: DMAADNH Encoding



Assembler syntax

`DMAADNH <address_register>, <16 -bit immediate>`

where:

`<address_register>` Selects the address register to use. It must be either:

`SAR SAR`*n* register and sets `ra` to 0

`DAR DAR`*n* register and sets `ra` to 1

`<16- bit immediate>` The immediate value to be added to the `<address_register>`.

Note: You should specify the 16-bit immediate as the number that is to be represented in the instruction encoding. For example, `DMAADNH DAR`, 0xFFF0 causes the value 0xFFFFFFF0 to be added to the current value of the Destination Address register, effectively subtracting 16 from the `DAR`.

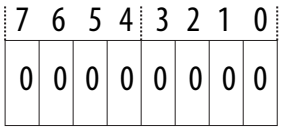
Operation

You can only use this instruction in a DMA channel thread.

DMAEND

End signals to the DMAC that the DMA sequence is complete. After all DMA transfers are complete for the DMA channel, the DMAC moves the channel to the Stopped state. It also flushes data from the MFIFO buffer and invalidates all cache entries for the thread.

Figure 16-10: DMAEND Instruction Encoding



Assembler syntax

`DMAEND`

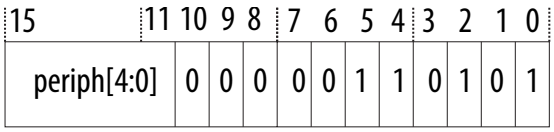
Operation

You can use the instruction with the DMA manager thread and the DMA channel thread.

DMAFLUSHP

Flush Peripheral clears the state in the DMAC that describes the contents of the peripheral and sends a message to the peripheral to resend its level status.

Figure 16-11: DMAFLUSHP Instruction Encoding



Assembler syntax

`DMAFLUSHP <peripheral>`

where:

`<peripheral>` 5-bit immediate, value 0-31

Operation

You can only use this instruction in a DMA channel thread.

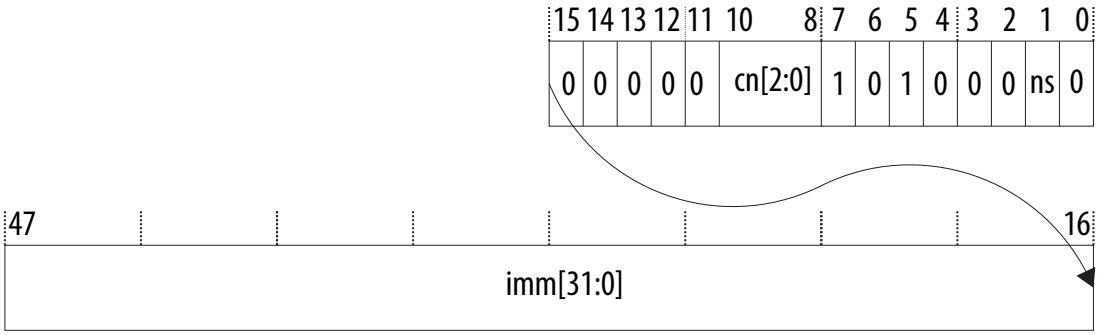
DMAGO

When the DMA manager executes Go for a DMA channel that is in the Stopped state, it performs the following steps on the DMA channel:

1. Moves a 32-bit immediate into the program counter
2. Sets its security state
3. Updates it to the Executing state.

Note: If a DMA channel is not in the Stopped state when the DMA manager executes `DMAGO` then the DMAC does not execute `DMAGO` but instead it executes `DMANOP`.

Figure 16-12: DMAGO Instruction Encoding



Assembler syntax

`DMAGO <channel_number>, <32- bit_immediate> [, ns]`

where:

`<channel_number>` Selects a DMA channel. It must be one of:

`C0` DMA channel 0

`C1` DMA channel 1

C2 DMA channel 2

C3 DMA channel 3

C4 DMA channel 4

C5 DMA channel 5

C6 DMA channel 6

C7 DMA channel 7

Note: If you provide a channel number that is not available for your configuration of the DMAC, the DMA manager thread aborts.

<32- bit_immediate> The immediate value that is written to the CPC*n* register for the selected <channel_number>.

[ns]

- If ns is present, the DMA channel operates in the Non-secure state.
- Otherwise, the execution of the instruction depends on the security state of the DMA manager:

DMA manager is in the Secure state—DMA channel operates in the Secure state.

DMA manager is in the Non-secure state—The DMAC aborts.

Operation

You can only use this instruction with the DMA manager thread.

DMAKILL

Kill instructs the DMAC to immediately terminate execution of a thread. Depending on the thread type, the DMAC performs the following steps:

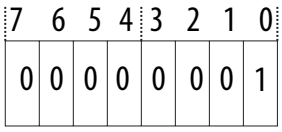
DMA Manager Thread

1. Invalidates all cache entries for the DMA manager.
2. Moves the DMA manager to the Stopped state.

DMA Channel Thread

1. Moves the DMA channel to the Killing state.
2. Waits for AXI transactions, with an ID equal to the DMA channel number, to complete.
3. Invalidates all cache entries for the DMA channel.
4. Remove all entries in the MFIFO buffer for the DMA channel.
5. Remove all entries in the read buffer queue and write buffer queue for the DMA channel.
6. Moves the DMA channel to the Stopped state.

Figure 16-13: DMAKILL Instruction Encoding



Assembler syntax

DMAKILL

Operation

You can use the instruction with the DMA manager thread and the DMA channel thread.

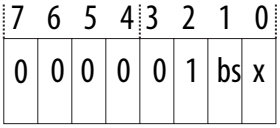
Note: You must not use the DMAKILL instruction in DMA channel programs. To issue a DMAKILL instruction, use the DBGINST0 register.

DMALD[S | B]

Load instructs the DMAC to perform a DMA load, using AXI transactions that the source address registers and channel control registers specify. It places the read data into the MFIFO buffer and tags it with the corresponding channel number. DMALD is an unconditional instruction but DMALDS and DMALDB are conditional on the state of the request_type flag. If the src_inc bit in the channel control registers is set to incrementing, the DMAC updates the source address registers after it executes DMALD[S|B].

Note: The DMAC sets the value of request_type when it executes a DMAWFP instruction.

Figure 16-14: DMALD[S|B] Instruction Encoding



Assembler syntax

DMALD[S|B]

where:

[S] If S is present, the assembler sets bs to 0 and x to 1. The instruction is conditional on the state of the request_type flag:

- request_type = Single

The DMAC performs a DMALD instruction and it sets arlen[3:0]=0x0 so that the AXI read transaction length is one. The DMAC ignores the value of the src_burst_len field in the channel control registers.

- request_type = Burst

The DMAC performs a DMANOP instruction. The DMAC increments the channel PC to the next instruction. No state change occurs.

[B] If B is present, the assembler sets bs to 1 and x to 1. The instruction is conditional on the state of the request_type flag:

- request_type = Single

The DMAC performs a DMANOP instruction. The DMAC increments the channel PC to the next instruction. No state change occurs.

- request_type = Burst

The DMAC performs a DMALD.

If you do not specify the S or B operand, the assembler sets bs to 0 and x to 0, and the DMAC always executes a DMA load.

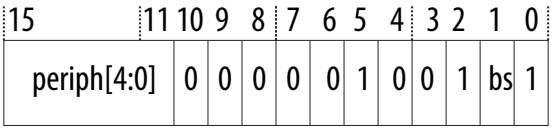
Operation

You can only use this instruction in a DMA channel thread. If you specify the S or B operand, execution of the instruction is conditional on the state of `request_type` matching that of the instruction.

DMALDP<S | B>

Load and notify Peripheral instructs the DMAC to perform a DMA load, using AXI transactions that source address registers and channel control registers specify. It places the read data into a FIFO buffer that is tagged with the corresponding channel number and after it receives the last data item, it sends an acknowledgement to the peripheral that the data transfer is complete. If the `src_inc` bit in the channel control registers is set to incrementing, the DMAC updates source address registers after it executes `DMALDP<S|B>`.

Figure 16-15: DMALDP<S|B> Instruction Encoding



Assembler syntax

```
DMALDP<S|B> <peripheral>
```

where:

`<S>` When S is present, the assembler sets `bs` to 0. The instruction is conditional on the state of the `request_type` flag:

- `request_type` = Single

The DMAC performs a `DMALDP` instruction and it sets `arlen[3:0]`=0x0 so that the AXI read transaction length is one. The DMAC ignores the value of the src_burst_len field in the channel control registers.

- `request_type` = Burst

The DMAC performs a `DMANOP`.

`<B>` When B is present, the assembler sets `bs` to 1. The instruction is conditional on the state of the `request_type` flag:

- `request_type` = Single

The DMAC performs a `DMANOP`.

- `request_type` = Burst

The DMAC performs a load using a burst DMA transfer.

`<peripheral>` 5-bit immediate, value 0-31.

Note: The DMAC sets the value of the `request_type` flag when it executes a `DMAWFP` instruction.

Operation

You can only use this instruction in a DMA channel thread. Execution of the instruction is conditional on the state of the `request_type` flag matching that of the instruction.

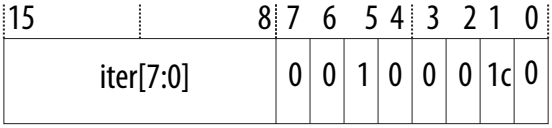
DMALP

Loop instructs the DMAC to load an 8-bit value into the loop counter register you specify.

This instruction indicates the start of a section of instructions, and you set the end of the section using the `DMALPEND` instruction. The DMAC repeats the set of instructions that you insert between `DMALP` and `DMALPEND` until the value in the loop counter register reaches zero.

Note: The DMAC saves the value of the PC for the instruction that follows `DMALP`. After the DMAC executes `DMALPEND`, and the loop counter register is not zero, this enables it to execute the first instruction in the loop.

Figure 16-16: DMALP Instruction Encoding



Assembler syntax

`DMALP <loop_iterations>`

where:

`<loop_iterations>`

Specifies the number of loops to perform, range 1-256.

- The assembler determines the loop counter register to use and either:
- Sets `lc` to 0, and the DMAC writes the value loop_iterations minus 1 to the loop counter 0 registers
- Sets `lc` to 1, and the DMAC writes the value loop_iterations minus 1 to the loop counter 1 registers.

Operation

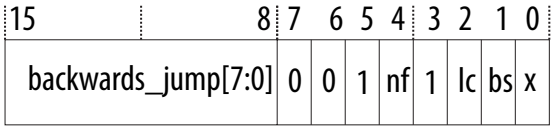
You can only use this instruction in a DMA channel thread.

DMALPEND[S | B]

Loop End indicates the last instruction in the program loop but the behavior of the DMAC depends on whether `DMALP` or `DMALPFE` starts the loop. If a loop starts with:

- `DMALP` The loop has a defined loop count and `DMALPEND[S|B]` instructs the DMAC to read the value of the loop counter register. If a loop counter register returns:
  - Zero—The DMAC executes a `DMANOP` and therefore exits the loop.
  - Nonzero—The DMAC decrements the value in the loop counter register and updates the thread PC to contain the address of the first instruction in the program loop, that is, the instruction that follows the `DMALP`.
- `DMALPFE` The loop has an undefined loop count and the DMAC uses the state of the `request_last` flag to control when it exits the loop. If the `request_last` flag is:
  - 0—The DMAC updates the thread PC to contain the address of the first instruction in the program loop, that is, the instruction that follows the `DMALP`.
  - 1—The DMAC executes a `DMANOP` and therefore exits the loop.

Figure 16-17: DMALPEND[S|B] Instruction Encoding



Assembler syntax

`DMALPEND[S|B]`

where:

`[S]` If S is present and the loop starts with `DMALP`, then the assembler sets `bs` to 0 and `x` to 1. The instruction is conditional on the state of the `request_type` flag:

- `request_type` = Single
  - The DMAC executes the `DMALPEND`
- `request_type` = Burst
  - The DMAC performs a `DMANOP` and therefore exits the loop.

`[B]` If B is present and the loop starts with `DMALP`, then the assembler sets `bs` to 1 and `x` to 1. The instruction is conditional on the state of the `request_type` flag:

- `request_type` = Single
  - The DMAC performs a `DMANOP` and therefore exits the loop.
- `request_type` = Burst
  - The DMAC executes the `DMALPEND`

If you do not specify the S or B operand, the assembler sets `bs` to 0 and `x` to 0, and the DMAC always executes the `DMALPEND`.

Note: You must not specify the S or B operand when a loop starts with `DMALPFE`. If you do, the assembler issues a warning message and sets `bs` to 0, `x` to 0, and `nf` to 1. In the same way as for `DMALPFE`, the DMAC uses the state of the `request_last` flag to control when it exits the loop.

Note: The DMAC sets the value of the:

- `request_type` flag when it executes a `DMAWFP` instruction.
- `request_last` flag to 1 when the corresponding peripheral issues the last request command through the peripheral request interface.

To correctly assign the additional bits in the `DMALPEND` instruction, the assembler determines the values for:

`backwards_jump[7:0]` Sets the relative location of the first instruction in the program loop. The assembler calculates the value for `backwards_jump[7:0]` by subtracting the address of the first instruction in the loop from the address of the `DMALPEND`.

- `nf` sets it to:
  - 0 if `DMALPFE` started the program loop
  - 1 if `DMALP` started the program loop.
- `lc` sets it to:
  - 0 if the loop counter 0 registers contains the loop counter value
  - 1 if the loop counter 1 registers contains the loop counter value
  - 1 if `DMALPFE` started the program loop.

Operation

You can only use this instruction in a DMA channel thread. If you specify the S or B operand, execution of the instruction is conditional on the state of the `request_type` flag matching that of the instruction.

Related Information

- **Peripheral Length Management** on page 16-17
- **DMAWFP** on page 16-51

DMALPFE

The assembler uses Loop Forever to configure certain bits in `DMALPEND`.

Note: When the assembler encounters `DMALPFE`, it does not create an instruction for the DMAC, but instead, it modifies the behavior of `DMALPEND`. The insertion of `DMALPFE` in program code identifies the start of the loop.

Assembler syntax

```
DMALPFE
```

Related Information

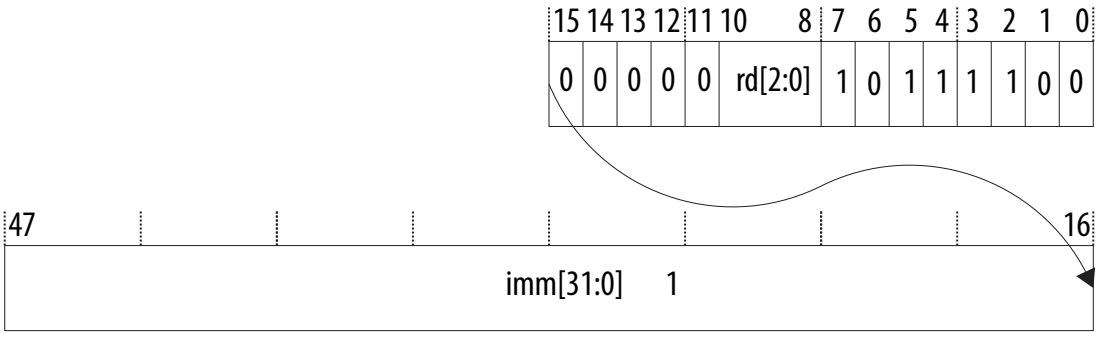
DMALPEND[S | B] on page 16-43

DMAMOV

Move instructs the DMAC to move a 32-bit immediate into the following registers:

- source address registers
- destination address registers
- channel control registers

Figure 16-18: DMAMOV Instruction Encoding



Assembler syntax

```
DMAMOV <destination_register>, <32- bit_immediate>
```

where:

`<destination_register>`

The valid registers are:

- `SAR`—selects the source address registers and sets `rd` to b000
- `CCR`—selects the channel control registers and sets `rd` to b001
- `DAR`—selects the destination address registers and sets `rd` to b010

`<32 -bit_immediate>`

A 32-bit value that is written to the specified destination register.

Note: For information about using the assembler to program the various fields that the channel control registers, refer to DMAMOV CCR section

Operation

You can only use this instruction in a DMA channel thread.

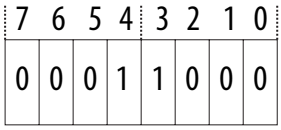
Related Information

DMAMOV CCR on page 16-53
Information about using the assembler to program the various fields that the channel control registers

DMANOP

No Operation does nothing. You can use this instruction for code alignment purposes.

Figure 16-19: DMANOP Instruction Encoding



Assembler syntax

```
DMANOP
```

Operation

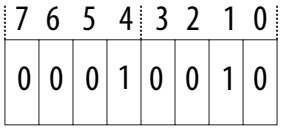
You can use the instruction with the DMA manager thread and the DMA channel thread.

DMARMB

Read Memory Barrier forces the DMA channel to wait until all of the executed `DMALD` instructions for that channel have been issued on the AXI master interface and have completed.

This enables write-after-read sequences to the same address location with no hazards.

Figure 16-20: DMARMB Instruction Encoding



Assembler syntax

```
DMARMB
```

Operation

You can only use this instruction in a DMA channel thread.

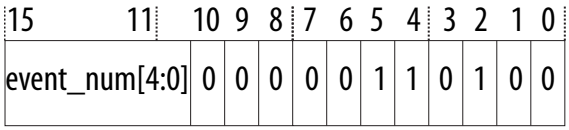
DMASEV

Send Event instructs the DMAC to modify an event-interrupt resource. Depending on how you program the interrupt enable register, this either:

- Generates event <event_num>

  Note: Typically, you use `DMAWFE` to stall a thread and then another thread executes `DMASEV`, using the appropriate event number, to unstall the waiting thread.
- Signals an interrupt using `irq <event_num>`.

Figure 16-21: DMASEV Instruction Encoding



Assembler syntax

```
DMASEV <event_num>
```

where:

`<event_num>` 5-bit immediate, value 0-31

Note: The DMAC aborts the thread if you select an event number that is not available.

Operation

You can use the instruction with the DMA manager thread and the DMA channel thread.

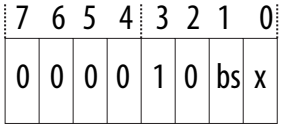
Related Information

- **Using Events and Interrupts** on page 16-22
- **Using an Event to Restart DMA Channels** on page 16-22

DMAST[S | B]

Store instructs the DMAC to transfer data from the FIFO buffer to the location that the destination address registers specifies, using AXI transactions that the DA register and channel control registers specify. If the `dst_inc` bit in the channel control registers is set to incrementing, the DMAC updates the destination address registers after it executes `DMAST[S|B]`.

Figure 16-22: DMAST[S|B] Instruction Encoding



Assembler syntax

`DMAST[S|B]`

where:

`[S]` If S is present, the assembler sets `bs` to 0 and `x` to 1. The instruction is conditional on the state of the `request_type` flag:

- `request_type` = Single
- The DMAC performs a `DMAST` instruction and it sets `awlen[3:0]`=0x0 so that the AXI write transaction length is one. The DMAC ignores the v value of the dst_burst_len field in the channel control registers.
- `request_type` = Burst
- The DMAC performs a `DMANOP` instruction. The DMAC increments the channel PC to the next instruction. No state change occurs.

`[B]` If B is present, the assembler sets `bs` to 1 and `x` to 1. The instruction is conditional on the state of the `request_type` flag:

- `request_type` = Single
- The DMAC performs a `DMANOP` instruction. The DMAC increments the channel PC to the next instruction. No state change occurs.
- `request_type` = Burst
- The DMAC performs a `DMAST`.
- If you do not specify the S or B operand, the assembler sets `bs` to 0 and `x` to 0, and the DMAC always executes a DMA store.

Note: The DMAC sets the value of the `request_type` flag when it executes a `DMAWFP` instruction.

Operation

You can only use this instruction in a DMA channel thread. If you specify the S or B operand, execution of the instruction is conditional on the state of the `request_type` flag matching that of the instruction.

The DMAC only commences the burst when the MFIFO buffer contains all of the data necessary to complete the burst transfer.

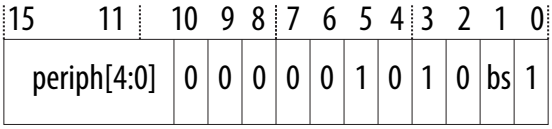
Related Information

DMAWFP on page 16-51

DMASTP<S | B>

Store and notify Peripheral instructs the DMAC to transfer data from the FIFO buffer to the location that the destination address registers specifies, using AXI transactions that the DA register and channel control registers specify. It uses the DMA channel number to access the appropriate location in the FIFO buffer. After the DMA store is complete, and the DMAC has received a buffered write response, it issues an acknowledgement to the peripheral that the data transfer is complete. If the `dst_inc` bit in the channel control registers is set to incrementing, the DMAC updates the destination address registers after it executes `DMASTP<S|B>`.

Figure 16-23: DMASTP<S|B> Instruction Encoding



Assembler syntax

`DMASTP<S|B> <peripheral>`

where:

`<S>` Sets `bs` to 0. This instructs the DMAC to perform:

- A single DMA store operation if `request_type` is programmed to Single
- The DMAC ignores the state of the dst_burst_len field in the channel control registers and always performs an AXI transfer with a burst length of one.
- A `DMANOP` if `request_type` is programmed to Burst.

`<B>` Sets `bs` to 1. This instructs the DMAC to perform:

- The DMA store if `request_type` is programmed to Burst
- A `DMANOP` if `request_type` is programmed to Single.

`<peripheral>` 5-bit immediate, value 0-31.

Note: The DMAC sets the value of the `request_type` flag when it executes a `DMAWFP` instruction.

Operation

You can only use this instruction in a DMA channel thread.

The DMAC only commences the burst when the MFIFO buffer contains all of the data necessary to complete the burst transfer.

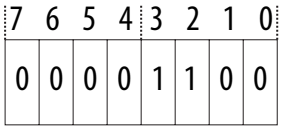
Related Information

DMAWFP on page 16-51

DMASTZ

Store Zero instructs the DMAC to store zeros, using AXI transactions that the destination address registers and channel control registers specify. If the `dst_inc` bit in the channel control registers is set to incrementing, the DMAC updates the destination address registers after it executes `DMASTZ`.

Figure 16-24: DMASTZ Instruction Encoding



Assembler syntax

```
DMASTZ
```

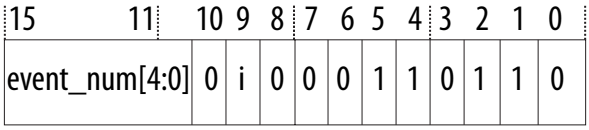
Operation

You can only use this instruction in a DMA channel thread.

DMAWFE

Wait For Event instructs the DMAC to halt execution of the thread until the event, that <event_num> specifies, occurs. When the event occurs, the thread moves to the Executing state and the DMAC clears the event.

Figure 16-25: DMAWFE Instruction Encoding



Assembler syntax

`DMAWFE <event_num>[, invalid]`

where:

`<event_num>` 5-bit immediate, value 0-31

`[invalid]` Sets `i` to 1. If `invalid` is present, the DMAC invalidates the instruction cache for the current DMA thread. If `invalid` is not present, then the assembler sets `i` to 0 and the DMAC does not invalidate the instruction cache for the current DMA.

- The DMAC aborts the thread if you select an event number that is not available for your configuration of the DMAC. To ensure cache coherency, you must use `invalid` when a processor writes the instruction stream for a DMA channel.

Operation

You can use the instruction with the DMA manager thread and the DMA channel thread.

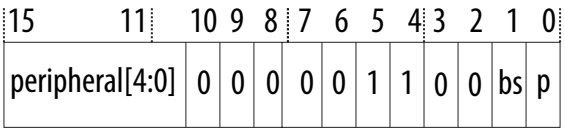
Related Information

Using Events and Interrupts on page 16-22

DMAWFP

Wait For Peripheral instructs the DMAC to halt execution of the thread until the specified peripheral signals a DMA request for that DMA channel.

Figure 16-26: DMAWFP Instruction Encoding



Assembler syntax

`DMAWFP <peripheral>, <single|burst|periph>`

where:

`<peripheral>` 5-bit immediate, value 0-31

Note: The DMAC aborts the thread if you select a peripheral number that is not available.

`<single>` Sets `bs` to 0 and `p` to 0. This instructs the DMAC to continue executing the DMA channel thread after it receives a single or burst DMA request. The DMAC sets the `request_type` to Single, for that DMA channel.

`<burst>` Sets `bs` to 1 and `p` to 0. This instructs the DMAC to continue executing the DMA channel thread after it receives a burst DMA request. The DMAC sets the `request_type` to Burst.

Note: The DMAC ignores single burst DMA requests.

`<periph>` Sets `bs` to 0 and `p` to 1. This instructs the DMAC to continue executing the DMA channel thread after it receives a single or burst DMA request. The DMAC sets the `request_type` to:

- **Single:** When it receives a single DMA request.
- **Burst:** When it receives a burst DMA request.

Operation

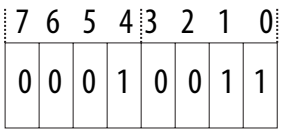
You can only use this instruction in a DMA channel thread.

DMAWMB

Write Memory Barrier forces the DMA channel to wait until all of the executed `DMAST` instructions for that channel have been issued on the AXI master interface and have completed.

This permits read-after-write sequences to the same address location with no hazards.

Figure 16-27: DMAWMB Instruction Encoding



Assembler syntax

```
DMAWMB
```

Operation

You can only use this instruction in a DMA channel thread.

Assembler Directives

The assembler provides several additional commands.

DCD

Assembler directive to place a 32-bit immediate in the instruction stream.

Syntax

DCD imm32

DCB

Assembler directive to place an 8-bit immediate in the instruction stream.

Syntax

DCB imm8

DMALP

Assembler directive to insert an iterative loop.

Syntax

DMALP [<LC0>|<LC1>] `<loop_iterations>`

where:

`<loop_iterations>`

An 8-bit value that specifies the number of loops to perform.

Note: For clarity in writing assembler instructions, the 8-bit value is the actual number of iterations of the loop to be executed. The assembler decrements this by one to create the actual value, 0-255, that the DMAC uses.

`[LC0]` If `LC0` is present, the DMAC stores `<loop_iterations>` in the loop counter 0 registers.

`[LC1]` If `LC1` is present, the DMAC stores `<loop_iterations>` in the loop counter 1 registers.

Note: If `LC0` or `LC1` is not present, the assembler determines the loop counter register to use.

DMALPFE

Assembler directive to insert a repetitive loop.

Syntax

DMALPFE

Enables the assembler to clear the `nf` bit that is present in `DMALPEND[S | B]`.

DMAMOV CCR

Assembler directive that enables you to program the channel control registers using the specified format.

Syntax

```
DMAMOV CCR,
[SB<1-16>] [SS<8|16|32|64|128>] [SA<I|F>]
[SP<imm3>] [SC<imm4>]
[DB<1-16>] [DS<8|16|32|64|128>] [DA<I|F>]
[DP<imm3>] [DC<imm4>]
[ES<8|16|32|64|128>]
```

Table 16-6: DMAMOV CCR Argument Description and Default Values

Syntax	Description	Options	Default
SA	Source address increment. Sets the value of `arburst[0]`	I = Increment F = Fixed	I
SS	Source burst size in bits. Sets the value of `arsize[2:0]`	8, 16, 32, or 64	8
SB	Source burst length. Sets the value of `arlen[3:0]`	1 to 16	1
SP	Source protection	0 to 7 (47)	0
SC	Source cache	0 to 15 (47)(48)	0
DA	Destination address increment. Sets the value of `awburst[0]`	I = Increment F = Fixed	I
DS	Destination burst size in bits. Sets the value of `awsize[2:0]`	8, 16, 32, or 64	8
DB	Destination burst length. Sets the value of `awlen[3:0]`	1 to 16	1
DP	Destination protection	0 to 7(47)	0
DC	Destination cache	0 to 15(47)(49)	0

(47) You must use decimal values when programming this immediate value.
(48) Because the DMAC ties ARCACHE[3] low, the assembler always sets bit 3 to 0 and uses bits [2:0] of your chosen value for SC.
(49) Because the DMAC ties AWCACHE[2] low, the assembler always sets bit 2 to 0 and uses bit [3] and bits [1:0] of your chosen value for DC.

Syntax	Description	Options	Default
ES	Endian swap size, in bits	8, 16, 32, or 64	8

MFIFO Buffer Usage Overview

About MFIFO Buffer Usage Overview

The MFIFO buffer is a shared resource that is utilized on a first-come, first-served basis by all currently active channels. To a program, it appears as a set of variable-depth parallel FIFO buffers, one per channel, with the restriction that the total depth of all the FIFOs cannot exceed the size of the MFIFO, which is 512 entries. The width of the AXI master interface is the same as the MFIFO buffer width.

The DMAC is capable of realigning data from the source to the destination. For example, the DMAC shifts the data by two byte lanes when it reads a word from address 0x103 and writes to address 0x205. All byte manipulations occur when data enters the MFIFO buffer, as a result of an AXI read due to a `DMALD` instruction, so that the DMAC does not need to manipulate the data when it removes it from the MFIFO buffer, as a result of an AXI write due to a `DMAST` instruction. Therefore the storage and packing of the data in the MFIFO buffer is determined by the destination address and transfer characteristics.

When a program specifies that incrementing transactions are to be performed to the destination, the DMAC packs data into the MFIFO buffer to minimize the usage of the MFIFO buffer entries. For example, the DMAC packs two 32-bit words into a single entry in the MFIFO buffer when the program uses a source address of 0x100, and destination address of 0x200.

The number of entries required to store the data loaded from a source is not a simple calculation of amount of source data divided by MFIFO buffer width in the following situations:

- The source address is not aligned to the AXI bus width.
- The destination address is not aligned to the AXI bus width.
- The transactions are to a fixed destination, that is, a non-incrementing address.

The `DMALD` and `DMAST` instructions each specify that an AXI transaction is to be performed. The amount of data transferred by an AXI transaction depends on the values programmed in to the `CCR` *n* register and the address of the transaction.

The following sections provide several example DMAC programs together with illustrations of the MFIFO buffer usage.

Note: These sections show MFIFO buffer usage in the following ways:

- A graph of the number of MFIFO buffer entries versus time
- A diagram of the byte-lane manipulation that the DMAC performs when data enters the MFIFO buffer.

Note: The numbers 0 and 7 in the MFIFO buffer diagrams indicate the byte lanes in the MFIFO buffer.

Related Information

ARM Information Center
Information about unaligned transfers available from the ARM Infocenter website.

Aligned Transfers

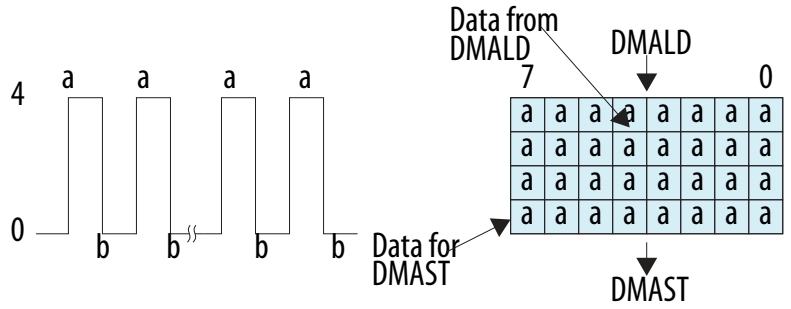
Simple Aligned Program

The following program shows the MFIFO buffer usage. In this program, the source address and destination address are aligned with the AXI data bus width.

```
DMAMOV CCR, SB4 SS64 DB4 DS64
DMAMOV SAR, 0x1000
DMAMOV DAR, 0x4000
DMALP 16
    DMALD    ; shown as a in figure below
               DMAST    ; shown as b in figure below
DMALPEND
DMAEND
```

Figure 16-28: Simple Aligned Program

Each `DMALD` requires four entries and each `DMAST` removes four entries.



This example has a static requirement of zero MFIFO buffer entries and a dynamic requirement of four MFIFO buffer entries.

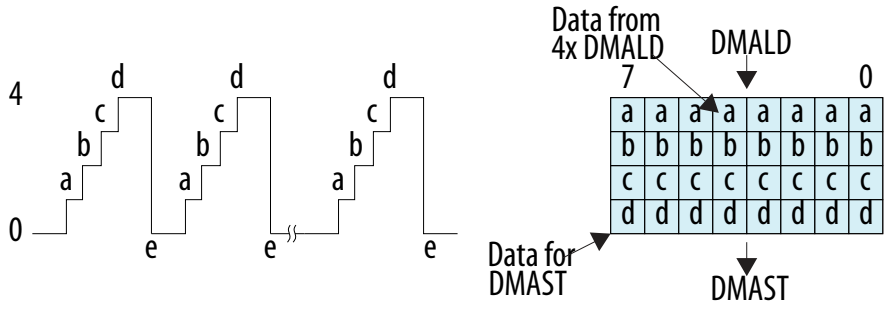
Aligned Asymmetric Program with Multiple Loads

The following program performs four loads for each store and the source address and destination address are aligned with the AXI data bus width.

```
DMAMOV CCR, SB1 SS64 DB4 DS64
DMAMOV SAR, 0x1000
DMAMOV DAR, 0x4000
DMALP 16
    DMALD    ; shown as a in the figure below
    DMALD ; shown as b in the figure below
    DMALD ; shown as c in the figure below
    DMALD ; shown as d in the figure below
    DMAST ; shown as e in the figure below
DMALPEND
DMAEND
```

Figure 16-29: Aligned Asymmetric Program with Multiple Loads

Each DMALD requires one entry and each DMAST removes four entries.



This example has a static requirement of zero MFIFO buffer entries and a dynamic requirement of four MFIFO buffer entries.

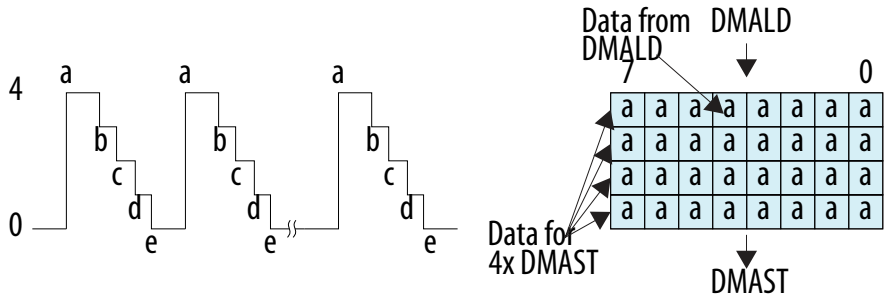
Aligned Asymmetric Program with Multiple Stores

The following program performs four stores for each load and the source address and destination address are aligned with the AXI data bus width.

```
DMAMOV CCR, SB4 SS64 DB1 DS64
DMAMOV SAR, 0x1000
DMAMOV DAR, 0x4000
DMALP 16
    DMALD ; shown as a in the figure below
    DMAST ; shown as b in the figure below
    DMAST ; shown as c in the figure below
    DMAST ; shown as d in the figure below
    DMAST ; shown as e in the figure below
DMALPEND
DMAEND
```

Figure 16-30: Aligned Asymmetric Program with Multiple Stores

Each DMALD requires four entries and each DMAST removes one entry.



This example has a static requirement of zero MFIFO buffer entries and a dynamic requirement of four MFIFO buffer entries.

Unaligned Transfers

Aligned Source Address to Unaligned Destination Address

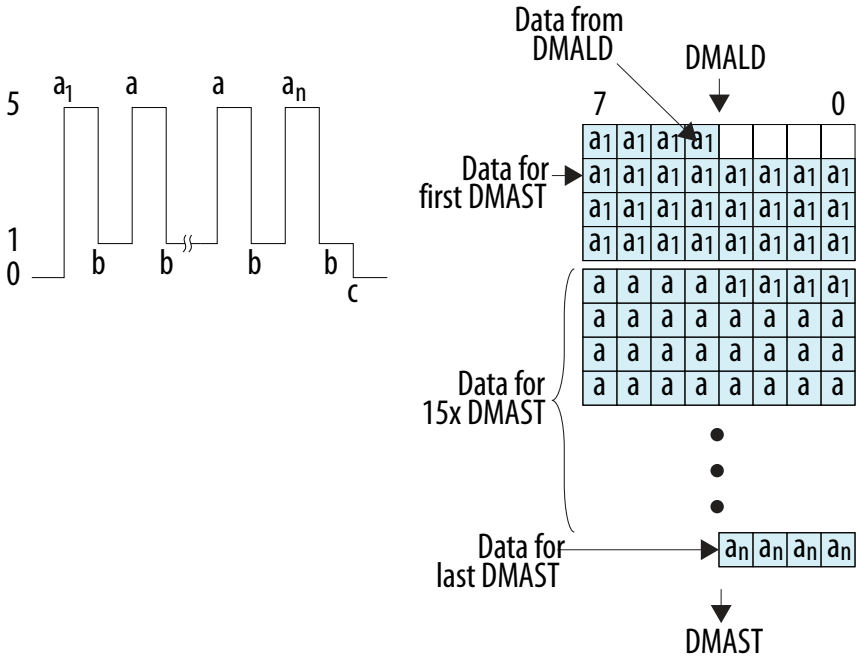
In the following program, the source address is aligned with the AXI data bus width but the destination address is unaligned. The destination address is not aligned to the destination burst size so the first DMAST instruction removes less data than the first DMALD instruction reads. Therefore, a final DMAST of a single word is required to clear the data from the MFIFO buffer.

```
DMAMOV CCR, SB4 SS64 DB4 DS64
DMAMOV SAR, 0x1000
DMAMOV DAR, 0x4004
DMALP 16
    DMALD ; shown as a1, ... a, an in the figure below
    DMAST ; shown as b in the figure below
DMALPEND
DMAMOV CCR, SB4 SS64 DB1 DS32
DMAST ; shown as c in the figure below
DMAEND
```

Figure 16-31: Aligned to Unaligned Program

The first DMALD instruction loads four doublewords but because the destination address is unaligned, the DMAC shifts them by four bytes and therefore uses five entries in the MFIFO buffer.

Each DMAST requires only four entries of data, and therefore the extra entry remains in use for the duration of the program, until it is emptied by the last DMAST.



This example has a static requirement of one MFIFO buffer entry and a dynamic requirement of four MFIFO buffer entries.

Unaligned Source Address to Aligned Destination Address

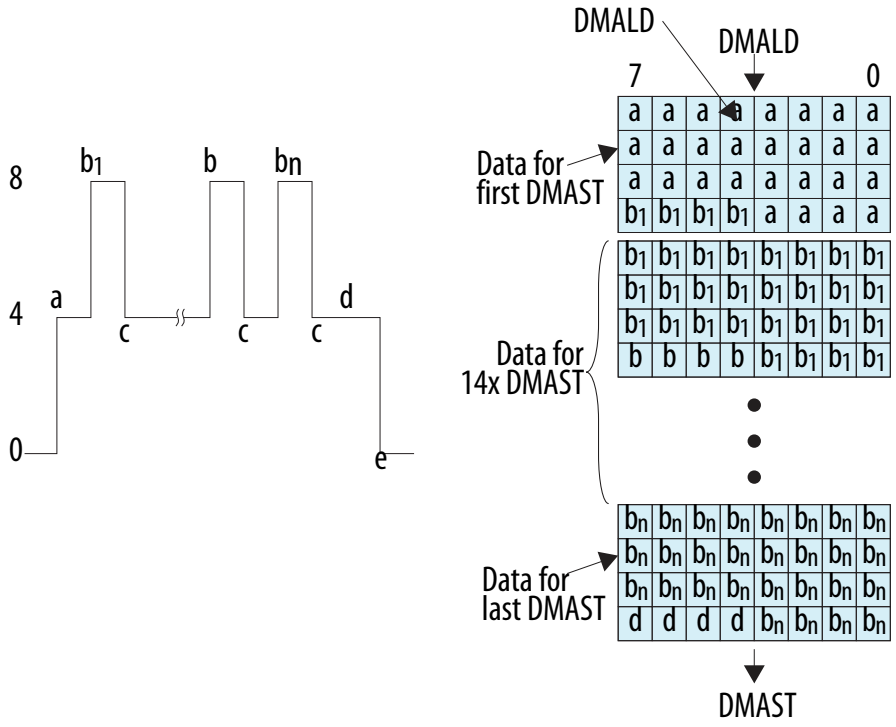
In this program, the source address is unaligned with the AXI data bus width but the destination address is aligned. The source address is not aligned to the source burst size so the first DMALD instruction reads in less data than the DMAST. Therefore, an extra DMALD is required to satisfy the first DMAST.

```
DMAMOV CCR, SB4 SS64 DB4 DS64
DMAMOV SAR, 0x1004
DMAMOV DAR, 0x4000
DMALD ; shown as a in the figure below
DMALP 15
    DMALD ; shown as b1, ... b, bn in the figure below
    DMAST ; shown as c in the figure below
DMALPEND
DMAMOV CCR, SB1 SS32 DB4 DS64
DMALD ; shown as d in the figure below
DMAST ; shown as e in the figure below
DMAEND
```

Figure 16-32: Unaligned to Aligned Program

The first DMALD instruction does not load sufficient data to enable the DMAC to execute a DMAST and therefore the program includes an additional DMALD, prior to the start of the loop.

After the first DMALD, the subsequent DMALDs align with the source burst size. This optimizes the **b** performance but it requires a larger number of MFIFO buffer entries.



Note: The DMALD shown as **d** does not increase the MFIFO buffer usage because it loads four bytes into an MFIFO buffer entry that the DMAC has already allocated to this channel.

This example has a static requirement of four MFIFO buffer entries and a dynamic requirement of four MFIFO buffer entries.

Unaligned Source Address to Aligned Destination Address with Excess Initial Load

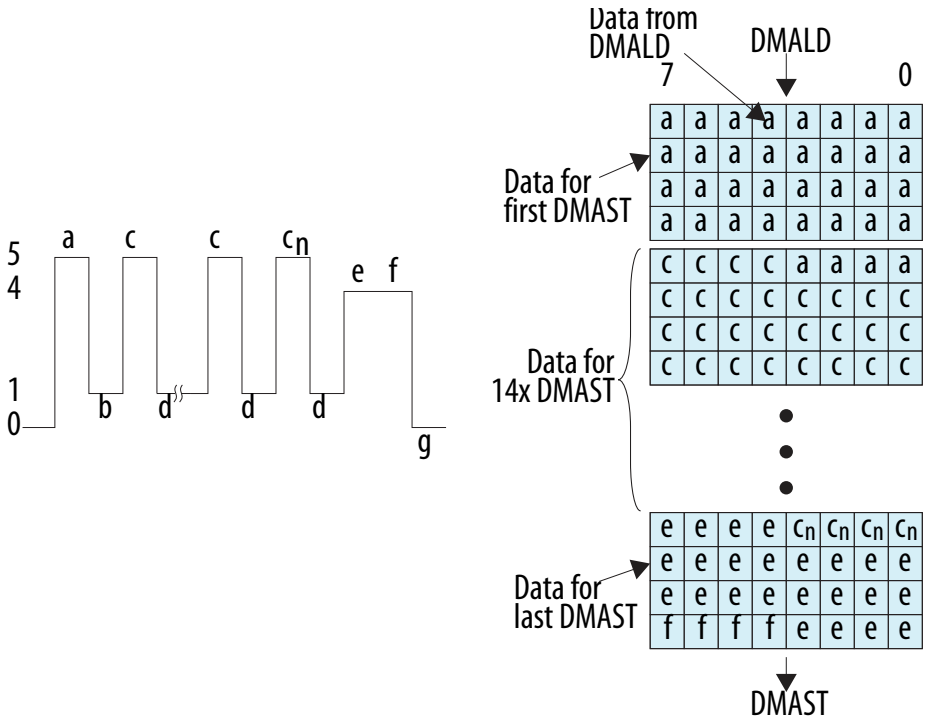
This program is an alternative to that described in *Unaligned Source Address to Aligned Destination Address*. The program executes a different sequence of source bursts that might be less efficient, but requires fewer MFIFO buffer entries.

```
DMAMOV CCR, SB5 SS64 DB4 DS64
DMAMOV SAR, 0x1004
DMAMOV DAR, 0x4000
DMALD ; shown as a in the figure below
DMAST ; shown as b in the figure below
DMAMOV CCR, SB4 SS64 DB4 DS64
DMALP 14
    DMALD ; shown as c and cn in the figure below
                DMAST ; shown as d in the figure below
DMALPEND
DMAMOV CCR, SB3 SS64 DB4 DS64
DMALD ; shown as e in the figure below
DMAMOV CCR, SB1 SS32 DB4 DS64
DMALD ; shown as f in the figure below
DMAST ; shown as g in the figure below
DMAEND
```

Figure 16-33: Unaligned to Aligned with Excess Initial Load

The first DMALD instruction loads five bursts of data to enable the DMAC to execute the first DMAST.

After the first DMALD, the subsequent DMALDs are not aligned to the source burst size, for example the second DMALD reads from address 0x1028. After the loop, the final two DMALDs read the data required to satisfy the final DMAST.



Note: The DMALD shown as **f** does not increase the MFIFO buffer usage because it loads four bytes into an MFIFO buffer entry that the DMAC has already allocated to this channel.

This example has a static requirement of one MFIFO buffer entry and a dynamic requirement of four MFIFO buffer entries.

Related Information

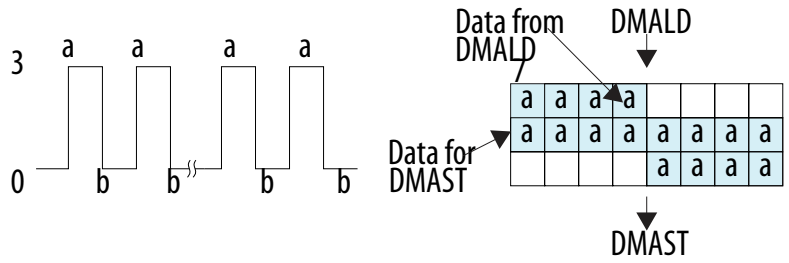
Unaligned Source Address to Aligned Destination Address on page 16-58

Aligned Burst Size Unaligned MFIFO Buffer

In this program, the destination address, which is narrower than the MFIFO buffer width, aligns with the burst size, but does not align with the MFIFO buffer width.

```
DMAMOV CCR, SB4 SS32 DB4 DS32
DMAMOV SAR, 0x1000
DMAMOV DAR, 0x4004
DMALP 16
 DMALD ; shown as a in the figure below
 DMAST ; shown as b in the figure below
DMALPEND
DMAEND
```

Figure 16-34: Aligned Burst with Unaligned MFIFO Buffer Width



In this example, the destination address is not 64-bit aligned, it requires three rather than the expected two MFIFO buffer entries.

This example has a static requirement of zero MFIFO buffer entries and a dynamic requirement of three MFIFO buffer entries.

Fixed Transfers

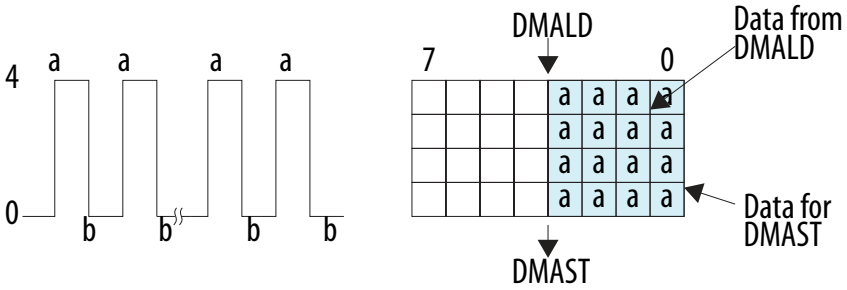
Fixed Destination with Aligned Address

In this program, the source address and destination address are aligned with the AXI data bus width, and the destination address is fixed.

```
DMAMOV CCR, SB2 SS64 DB4 DS32 DAF
DMAMOV SAR, 0x1000
DMAMOV DAR, 0x4000
DMALP 16
    DMALD ; shown as a in the figure below
    DMAST ; shown as b in the figure below
DMALPEND
DMAEND
```

Figure 16-35: Fixed Destination with Aligned Address

Each `DMALD` in the program loads two 64-bit data transfers into the MFIFO buffer. Because the destination address is a 32-bit fixed address, then the DMAC splits each 64-bit data item across two entries in the MFIFO buffer.

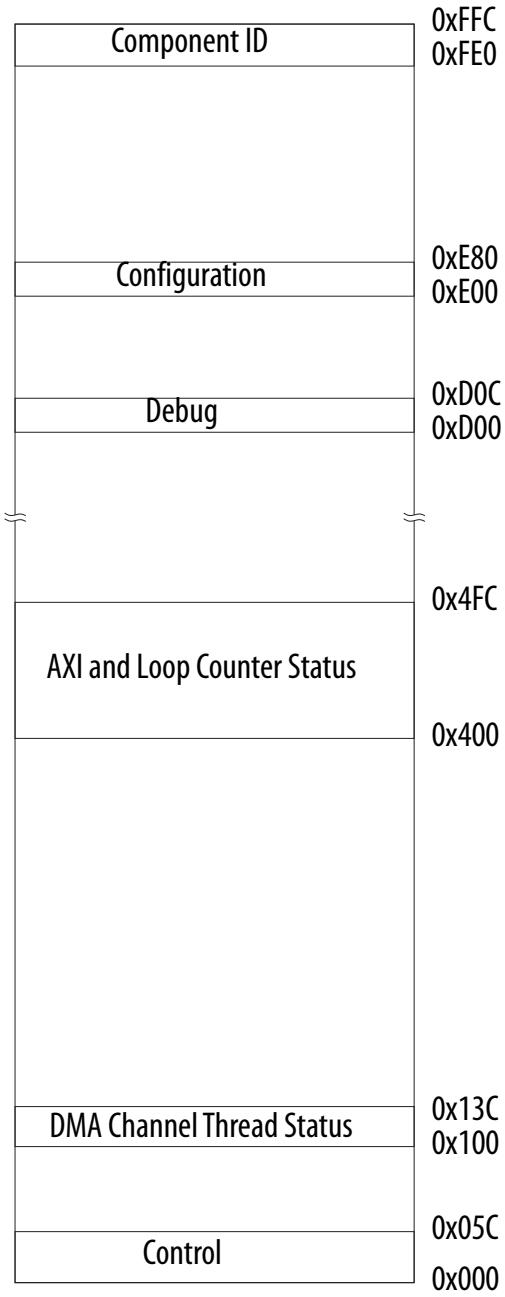


This example has a static requirement of zero MFIFO buffer entries and a dynamic requirement of four MFIFO buffer entries.

DMA Controller Address Map and Register Definitions

The following DMAC register map spans a 4 KB region, consists of the following sections.

Figure 16-36: DMAC Summary Register Map



- **Control registers—** allow you to control the DMAC.
- **DMA channel thread status registers—**provide the status of the DMA channel threads.
- **AXI and loop counter status registers—**provide the AXI transfer status and the loop counter status for each DMA channel thread.
- **Debug registers—**enable the following functionality:
  - Allow you to send instructions to a thread when debugging the program code.
  - Allow system firmware to send instructions to the DMA manager thread.
- **Configuration registers—**enable system firmware to identify the configuration of the DMAC and control the behavior of the watchdog.
- **Component ID registers—** enable system firmware to identify peripherals. Do not attempt to access reserved or unused address locations. Attempting to access these locations can result in an unpredictable behavior.

Related Information

Issuing Instructions to the DMAC using a Slave Interface on page 16-10

DMA Controller Address Map and Register Definitions

For complete HPS address map and register definitions, refer to "Arria 10 HPS Address Map and Register Definitions".

Related Information

Error Checking and Correction Controller on page 11-1
For more information about how to program the ECC registers, refer to this chapter.

dmac_ecc Address Map

Module Instance	Base Address	End Address
ecc_dmac_ecc_registerBlock	0xFF8C8000	0xFF8C83FF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
IP_REV_ID on page 11-282	0x0	32	RO	0x0	
CTRL on page 11-283	0x8	32	RW	0x0	ECC Control Register
INITSTAT on page 11-284	0xC	32	RW	0x0	This bit is used to set the initialize the memory and ecc to a known value

Register	Offset	Width	Access	Reset Value	Description
ERRINTEN on page 11-285	0x10	32	RW	0x0	Error Interrupt enable
ERRINTENS on page 11-286	0x14	32	RW	0x0	Error Interrupt set
ERRINTENR on page 11-286	0x18	32	RW	0x0	Error Interrupt reset.
INTMODE on page 11-287	0x1C	32	RW	0x0	Reads reflect SERRINTEN.
INTSTAT on page 11-288	0x20	32	RW	0x0	This bit is used to enable interrupt generation on SERR lookup table overflow. When all the entries in the table are valid=1 and this is bit is enabled, serr_req signal will be asserted.
INTTEST on page 11-289	0x24	32	RW	0x0	This bits is used to test interrupt from ECC RAM to GIC
MODSTAT on page 11-290	0x28	32	RW	0x0	Counter feature status flag
DERRADDRA on page 11-290	0x2C	32	RO	0x0	This register shows the address of PORTA current double-bit error. RAM size will determine the maximum number of address bits.
SERRADDRA on page 11-291	0x30	32	RO	0x0	This register shows the address of PORTA current single-bit error. RAM size will determine the maximum number of address bits.
SERRCNTREG on page 11-292	0x3C	32	RW	0x0	Maximum counter value for single-bit error interrupt
ECC_Addrbus on page 11-293	0x40	32	RW	0x0	MSB bit of address is determined by ADR.

Register	Offset	Width	Access	Reset Value	Description
ECC_RData0bus on page 11-293	0x44	32	RO	0x0	Data will be read to this register field.
ECC_RData1bus on page 11-294	0x48	32	RO	0x0	Data will be read to this register field.
ECC_RData2bus on page 11-295	0x4C	32	RO	0x0	Data will be read to this register field.
ECC_RData3bus on page 11-295	0x50	32	RO	0x0	Data will be read to this register field.
ECC_WData0bus on page 11-296	0x54	32	WO	0x0	Data from the register will be written to the RAM.
ECC_WData1bus on page 11-297	0x58	32	WO	0x0	Data from the register will be written to the RAM.
ECC_WData2bus on page 11-297	0x5C	32	WO	0x0	Data from the register will be written to the RAM.
ECC_WData3bus on page 11-298	0x60	32	WO	0x0	Data from the register will be written to the RAM.
ECC_RDataecc0bus on page 11-299	0x64	32	RO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.
ECC_RDataecc1bus on page 11-299	0x68	32	RO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Register	Offset	Width	Access	Reset Value	Description
ECC_WDataecc0bus on page 11-300	0x6C	32	WO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.
ECC_WDataecc1bus on page 11-301	0x70	32	WO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.
ECC_dbytectrl on page 11-302	0x74	32	RW	0x0	Max number of implemented byte enabled is DAT/8
ECC_accctrl on page 11-303	0x78	32	RW	0x0	These bits determine which byte of data/ecc to write to RAM.
ECC_startacc on page 11-304	0x7C	32	RW	0x0	These bits determine which byte of data/ecc to write to RAM.
ECC_wdctrl on page 11-305	0x80	32	RW	0x0	Bits to Enable/Disable Watch Dog Timer
SERRLKUPA0 on page 11-305	0x90	32	RW	0x0	Single-bit error address in LOOKUP TABLE for PORTA.

dmac_ecc Summary

Base Address: 0xFF8C8000

Register Address Offset	Bit Fields
ecc_dmac_ecc_register-Block	

Register Address Offset	Bit Fields															
IP_REV_ID 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SIREV RO 0x0															
CTRL 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															INITA 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							CNT_RSTA 0x0	Reserved							ECC_EN 0x0
INITSTAT 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															INITCOM-PLETEA 0x0
ERRINTEN 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTEN 0x0
ERRINTENS 0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTS 0x0
ERRINTENR 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTR 0x0

Register Address Offset	Bit Fields															
INTMODE 0x1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															INTONCMP 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							INTONOVF 0x0	Reserved							INTMODE 0x0
INTSTAT 0x20	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							DERRPENA 0x0	Reserved							SERRPENA 0x0
INTTEST 0x24	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							TDERRA 0x0	Reserved							TSERRA 0x0
MODSTAT 0x28	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															CMPFLGA 0x0
DERRADDRA 0x2C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							Address 0x0								
SERRADDRA 0x30	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							Address 0x0								

Register Address Offset	Bit Fields															
SERRCNTREG 0x3C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	SERRCNT 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SERRCNT 0x0															
ECC_Addrbus 0x40	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							ECC_AddrBUS 0x0								
ECC_RData0bus 0x44	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_RData1bus 0x48	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_RData2bus 0x4C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_RData3bus 0x50	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															

Register Address Offset	Bit Fields															
ECC_WData0bus 0x54	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_WData1bus 0x58	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_WData2bus 0x5C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_WData3bus 0x60	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_RDataecc0bus 0x64	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved			ECC_RDataecc3BUS 0x0					Reserved			ECC_RDataecc2BUS 0x0				
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved			ECC_RDataecc1BUS 0x0					Reserved			ECC_RDataecc0BUS 0x0				
ECC_RDataecc1bus 0x68	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved			ECC_RDataecc7BUS 0x0					Reserved			ECC_RDataecc6BUS 0x0				
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved			ECC_RDataecc5BUS 0x0					Reserved			ECC_RDataecc4BUS 0x0				

Register Address Offset	Bit Fields															
ECC_WDataecc0bus 0x6C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved			ECC_WDataecc3BUS 0x0					Reserved			ECC_WDataecc2BUS 0x0				
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved			ECC_WDataecc1BUS 0x0					Reserved			ECC_WDataecc0BUS 0x0				
ECC_WDataecc1bus 0x70	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved			ECC_WDataecc7BUS 0x0					Reserved			ECC_WDataecc6BUS 0x0				
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved			ECC_WDataecc5BUS 0x0					Reserved			ECC_WDataecc4BUS 0x0				
ECC_dbytectrl 0x74	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								DBEN 0x0							
ECC_accctrl 0x78	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							RDWR 0x0	Reserved						ECCOVR 0x0	DATAOVR 0x0
ECC_startacc 0x7C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															ENBUSA 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															
ECC_wdctrl 0x80	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															WDEN_RAM 0x0

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SERRLKUPA0 0x90	VALID RW 0x0	Reserved														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							Address RO 0x0								

IP_REV_ID

Module Instance	Base Address	Register Address
ecc_dmac_ecc_registerBlock	0xFF8C8000	0xFF8C8000

Offset: 0x0

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SIREV RO 0x0															

IP_REV_ID Fields

Bit	Name	Description	Access	Reset
15:0	SIREV	IP Rev # These bits indicate the silicon revision number.	RO	0x0

CTRL

ECC Control Register

Module Instance	Base Address	Register Address
ecc_dmac_ecc_registerBlock	0xFF8C8000	0xFF8C8008

Offset: 0x8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															INITA 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							CNT_RSTA 0x0	Reserved							ECC_EN 0x0

CTRL Fields

Bit	Name	Description	Access	Reset
16	INITA	Enable for the hardware memory initialization PORTA.	RW	0x0
8	CNT_RSTA	Enable to reset internal single-bit error counter A value to zero	RW	0x0
0	ECC_EN	Enable for the ECC detection and correction logic.	RW	0x0

INITSTAT

This bit is used to set the initialize the memory and ecc to a known value

Module Instance	Base Address	Register Address
ecc_dmac_ecc_registerBlock	0xFF8C8000	0xFF8C800C

Offset: 0xC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															INITCOMPLETEA 0x0

INITSTAT Fields

Bit	Name	Description	Access	Reset
0	INITCOMPLETEA	This bit is used to verify if the hardware memory initialization has completed PORTB.	RW	0x0

ERRINTEN

Error Interrupt enable

Module Instance	Base Address	Register Address
ecc_dmac_ecc_registerBlock	0xFF8C8000	0xFF8C8010

Offset: 0x10

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															SERRINTEN 0x0

ERRINTEN Fields

Bit	Name	Description	Access	Reset
0	SERRINTEN	This bit is used to enable the single bit error interrupt of ECC RAM system	RW	0x0

ERRINTENS

Error Interrupt set

Module Instance	Base Address	Register Address
ecc_dmac_ecc_registerBlock	0xFF8C8000	0xFF8C8014

Offset: 0x14

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															SERRINTS 0x0

ERRINTENS Fields

Bit	Name	Description	Access	Reset
0	SERRINTS	This bit is used to set the single-bit error interrupt bit.	RW	0x0

ERRINTENR

Error Interrupt reset.

Module Instance	Base Address	Register Address
ecc_dmac_ecc_registerBlock	0xFF8C8000	0xFF8C8018

Offset: 0x18

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															SERRINTR 0x0

ERRINTENR Fields

Bit	Name	Description	Access	Reset
0	SERRINTR	This bit is used to reset the single-bit error interrupt bit. Reads reflect SERRINTEN. 1'b0: Writing of zero has no effect. 1'b1: By writing one, this bit will reset SERRINTEN bit to 0. This is performing a bitwise writing of this feature.	RW	0x0

INTMODE

Reads reflect SERRINTEN.

Module Instance	Base Address	Register Address
ecc_dmac_ecc_registerBlock	0xFF8C8000	0xFF8C801C

Offset: 0x1C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															INTONCMP 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							INTONOVF 0x0	Reserved							INTMODE 0x0

INTMODE Fields

Bit	Name	Description	Access	Reset
16	INTONCMP	Enable interrupt on compare.	RW	0x0
8	INTONOVF	Enable interrupt on overflow.	RW	0x0
0	INTMODE	Interrupt mode for single-bit errors.	RW	0x0

INTSTAT

This bit is used to enable interrupt generation on SERR lookup table overflow. When all the entries in the table are valid=1 and this is bit is enabled, serr_req signal will be asserted.

Module Instance	Base Address	Register Address
ecc_dmac_ecc_registerBlock	0xFF8C8000	0xFF8C8020

Offset: 0x20

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							DERRPENA 0x0	Reserved							SERRPENA 0x0

INTSTAT Fields

Bit	Name	Description	Access	Reset
8	DERRPENA	Double-bit error pending for PORTA.	RW	0x0
0	SERRPENA	Single-bit error pending for PORTA.	RW	0x0

INTTEST

This bits is used to test interrupt from ECC RAM to GIC

Module Instance	Base Address	Register Address
ecc_dmac_ecc_registerBlock	0xFF8C8000	0xFF8C8024

Offset: 0x24

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							TDERRA 0x0	Reserved							TSERRA 0x0

INTTEST Fields

Bit	Name	Description	Access	Reset
8	TDERRA	Test PORTA Double-bit error.	RW	0x0
0	TSERRA	Test PORTA Single-bit error.	RW	0x0

MODSTAT

Counter feature status flag

Module Instance	Base Address	Register Address
ecc_dmac_ecc_registerBlock	0xFF8C8000	0xFF8C8028

Offset: 0x28

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															CMPFLGA 0x0

MODSTAT Fields

Bit	Name	Description	Access	Reset
0	CMPFLGA	Port A compare status flag	RW	0x0

DERRADDRA

This register shows the address of PORTA current double-bit error. RAM size will determine the maximum number of address bits.

Module Instance	Base Address	Register Address
ecc_dmac_ecc_registerBlock	0xFF8C8000	0xFF8C802C

Offset: 0x2C

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							Address 0x0								

DERRADDRA Fields

Bit	Name	Description	Access	Reset
8:0	Address	Recent double-bit error address.	RO	0x0

SERRADDRA

This register shows the address of PORTA current single-bit error. RAM size will determine the maximum number of address bits.

Module Instance	Base Address	Register Address
ecc_dmac_ecc_registerBlock	0xFF8C8000	0xFF8C8030

Offset: 0x30

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							Address 0x0								

SERRADDRA Fields

Bit	Name	Description	Access	Reset
8:0	Address	Recent single-bit error address.	RO	0x0

SERRCNTREG

Maximum counter value for single-bit error interrupt

Module Instance	Base Address	Register Address
ecc_dmac_ecc_registerBlock	0xFF8C8000	0xFF8C803C

Offset: 0x3C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SERRCNT 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SERRCNT 0x0															

SERRCNTREG Fields

Bit	Name	Description	Access	Reset
31:0	SERRCNT	Counter value	RW	0x0

ECC_Addrbus

MSB bit of address is determined by ADR.

Module Instance	Base Address	Register Address
ecc_dmac_ecc_registerBlock	0xFF8C8000	0xFF8C8040

Offset: 0x40

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							ECC_AddrBUS 0x0								

ECC_Addrbus Fields

Bit	Name	Description	Access	Reset
8:0	ECC_AddrBUS	Address will be driven to RAM to either read or write the data. Address will be latched by the RAM when the Enbus is asserted.	RW	0x0

ECC_RData0bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_dmac_ecc_registerBlock	0xFF8C8000	0xFF8C8044

Offset: 0x44

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData0bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_RDataBUS	ECC_RDataBUS[31:0].	RO	0x0

ECC_RData1bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_dmac_ecc_registerBlock	0xFF8C8000	0xFF8C8048

Offset: 0x48

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData1bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_RDataBUS	ECC_RDataBUS[63:32].	RO	0x0

ECC_RData2bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_dmac_ecc_registerBlock	0xFF8C8000	0xFF8C804C

Offset: 0x4C

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData2bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_RDataBUS	ECC_RDataBUS[95:64].	RO	0x0

ECC_RData3bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_dmac_ecc_registerBlock	0xFF8C8000	0xFF8C8050

Offset: 0x50

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData3bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_RDataBUS	ECC_RDataBUS[127-96].	RO	0x0

ECC_WData0bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_dmac_ecc_registerBlock	0xFF8C8000	0xFF8C8054

Offset: 0x54

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData0bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_WDataBUS	ECC_WDataBUS[31:0].	WO	0x0

ECC_WData1bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_dmac_ecc_registerBlock	0xFF8C8000	0xFF8C8058

Offset: 0x58

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData1bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_WDataBUS	ECC_WDataBUS[63:32].	WO	0x0

ECC_WData2bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_dmac_ecc_registerBlock	0xFF8C8000	0xFF8C805C

Offset: 0x5C

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData2bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_WDataBUS	ECC_WDataBUS[95-64].	WO	0x0

ECC_WData3bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_dmac_ecc_registerBlock	0xFF8C8000	0xFF8C8060

Offset: 0x60

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData3bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_WDataBUS	ECC_WDataBUS[127-96].	WO	0x0

ECC_RDataecc0bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_dmac_ecc_registerBlock	0xFF8C8000	0xFF8C8064

Offset: 0x64

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved			ECC_RDataecc3BUS 0x0					Reserved			ECC_RDataecc2BUS 0x0				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			ECC_RDataecc1BUS 0x0					Reserved			ECC_RDataecc0BUS 0x0				

ECC_RDataecc0bus Fields

Bit	Name	Description	Access	Reset
28:24	ECC_RDataecc3BUS	Eccdata will be read to this register field.	RO	0x0
20:16	ECC_RDataecc2BUS	Eccdata will be read to this register field.	RO	0x0
12:8	ECC_RDataecc1BUS	Eccdata will be read to this register field.	RO	0x0
4:0	ECC_RDataecc0BUS	Eccdata will be read to this register field.	RO	0x0

ECC_RDataecc1bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_dmac_ecc_registerBlock	0xFF8C8000	0xFF8C8068

Offset: 0x68

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved			ECC_RDataecc7BUS 0x0					Reserved			ECC_RDataecc6BUS 0x0				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			ECC_RDataecc5BUS 0x0					Reserved			ECC_RDataecc4BUS 0x0				

ECC_RDataecc1bus Fields

Bit	Name	Description	Access	Reset
28:24	ECC_RDataecc7BUS	Eccdata will be read to this register field.	RO	0x0
20:16	ECC_RDataecc6BUS	Eccdata will be read to this register field.	RO	0x0
12:8	ECC_RDataecc5BUS	Eccdata will be read to this register field.	RO	0x0
4:0	ECC_RDataecc4BUS	Eccdata will be read to this register field.	RO	0x0

ECC_WDataecc0bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_dmac_ecc_registerBlock	0xFF8C8000	0xFF8C806C

Offset: 0x6C

Access: WO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved			ECC_WDataecc3BUS 0x0					Reserved			ECC_WDataecc2BUS 0x0				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			ECC_WDataecc1BUS 0x0					Reserved			ECC_WDataecc0BUS 0x0				

ECC_WDataecc0bus Fields

Bit	Name	Description	Access	Reset
28:24	ECC_WDataecc3BUS	Eccdata from the register will be written to the RAM.	WO	0x0
20:16	ECC_WDataecc2BUS	Eccdata from the register will be written to the RAM.	WO	0x0
12:8	ECC_WDataecc1BUS	Eccdata from the register will be written to the RAM.	WO	0x0
4:0	ECC_WDataecc0BUS	Eccdata from the register will be written to the RAM.	WO	0x0

ECC_WDataecc1bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_dmac_ecc_registerBlock	0xFF8C8000	0xFF8C8070

Offset: 0x70

Access: WO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved			ECC_WDataecc7BUS 0x0					Reserved			ECC_WDataecc6BUS 0x0				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			ECC_WDataecc5BUS 0x0					Reserved			ECC_WDataecc4BUS 0x0				

ECC_WDataecc1bus Fields

Bit	Name	Description	Access	Reset
28:24	ECC_WDataecc7BUS	Eccdata from the register will be written to the RAM.	WO	0x0
20:16	ECC_WDataecc6BUS	Eccdata from the register will be written to the RAM.	WO	0x0
12:8	ECC_WDataecc5BUS	Eccdata from the register will be written to the RAM.	WO	0x0
4:0	ECC_WDataecc4BUS	Eccdata from the register will be written to the RAM.	WO	0x0

ECC_dbytectrl

ECC Data Byte Control. The maximum number of implemented byte enables equals the data width of the RAM divided by 8.

Module Instance	Base Address	Register Address
ecc_dmac_ecc_registerBlock	0xFF8C8000	0xFF8C8074

Offset: 0x74

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								DBEN 0x0							

ECC_dbytectrl Fields

Bit	Name	Description	Access	Reset
7:0	DBEN	Byte or word enable for access.	RW	0x0

ECC_accctrl

These bits determine which byte of data/ecc to write to RAM.

Module Instance	Base Address	Register Address
ecc_dmac_ecc_registerBlock	0xFF8C8000	0xFF8C8078

Offset: 0x78

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							RDWR 0x0	Reserved						ECCOVR 0x0	DATAOVR 0x0

ECC_accctrl Fields

Bit	Name	Description	Access	Reset
8	RDWR	Control for read/write.	RW	0x0
1	ECCOVR	ECC Data Override.	RW	0x0
0	DATAOVR	RAM Data Override. Override the ECC_dataBUS register with RAM data in read mode set by ECC_RW. 1'b0: Data override disabled. 1'b1: Data override enabled.	RW	0x0

ECC_startacc

These bits determine which byte of data/ecc to write to RAM.

Module Instance	Base Address	Register Address
ecc_dmac_ecc_registerBlock	0xFF8C8000	0xFF8C807C

Offset: 0x7C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															ENBUSA 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															

ECC_startacc Fields

Bit	Name	Description	Access	Reset
16	ENBUSA	Start RAM access for PORTA.	RW	0x0

ECC_wdctrl

Bits to Enable/Disable Watch Dog Timer

Module Instance	Base Address	Register Address
ecc_dmac_ecc_registerBlock	0xFF8C8000	0xFF8C8080

Offset: 0x80

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															WDEN_RAM 0x0

ECC_wdctrl Fields

Bit	Name	Description	Access	Reset
0	WDEN_RAM	Enable watchdog timeout for OCP register access to IP RAM.	RW	0x0

SERRLKUPA0

Single-bit error address in LOOKUP TABLE for PORTA.

Module Instance	Base Address	Register Address
ecc_dmac_ecc_registerBlock	0xFF8C8000	0xFF8C8090

Offset: 0x90

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VALID RW 0x0	Reserved														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							Address RO 0x0								

SERRLKUPA0 Fields

Bit	Name	Description	Access	Reset
31	VALID	Valid flag bit. Valid bit indicates if the address in this register is current or stale.	RW	0x0
8:0	Address	Recent Single-bit error address. This register shows the address of the each single-bit error. RAM size will determine the maximum number of address bits. If ram size is 32 Kbytes, bit 30-16 will be reserved and read as zero.	RO	0x0

dma_NSCONTROL Address Map

This address space is allocated for DMA control registers.

Module Instance	Base Address	End Address
i_dma_NSCONTROL	0xFFDA0000	0xFFDA00FF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

dma_NSCHANNELSTATUS Address Map

This address space is allocated for the registers that provide the status of the DMA channel threads.

Module Instance	Base Address	End Address
i_dma_NSCHANNELSTATUS	0xFFDA0100	0xFFDA03FF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

dma_NSAXISTATUS Address Map

This address space is allocated for the registers that provide the AXI transfer status and loop counter status for each DMA channel thread.

Module Instance	Base Address	End Address
i_dma_NSAXISTATUS	0xFFDA0400	0xFFDA0CFF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

dma_NSDEBUG Address Map

This address space is allocated for the debug registers.

Module Instance	Base Address	End Address
i_dma_NSDEBUG	0xFFDA0D00	0xFFDA0DFF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

dma_NSCONFIG Address Map

This address space holds registers that can be read for configuration data and control watchdog behavior.

Module Instance	Base Address	End Address
i_dma_NSCONFIG	0xFFDA0E00	0xFFDA0FDF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

dma_NSCOMPID Address Map

This address space holds peripheral ID information.

Module Instance	Base Address	End Address
i_dma_NSCOMPID	0xFFDA0FE0	0xFFDA0FFF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

dma_SCONTROL Address Map

This address space is allocated for DMA control registers.

Module Instance	Base Address	End Address
i_dma_SCONTROL	0xFFDA1000	0xFFDA10FF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

dma_SCHANNELSTATUS Address Map

This address space is allocated for the registers that provide the status of the DMA channel threads.

Module Instance	Base Address	End Address
i_dma_SCHANNELSTATUS	0xFFDA1100	0xFFDA13FF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

dma_SAXISTATUS Address Map

This address space is allocated for the registers that provide the AXI transfer status and loop counter status for each DMA channel thread.

Module Instance	Base Address	End Address
i_dma_SAXISTATUS	0xFFDA1400	0xFFDA1CFF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

dma_SDEBUG Address Map

This address space is allocated for the debug registers.

Module Instance	Base Address	End Address
i_dma_SDEBUG	0xFFDA1D00	0xFFDA1DFF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

dma_SCONFIG Address Map

This address space holds registers that can be read for configuration data and control watchdog behavior.

Module Instance	Base Address	End Address
i_dma_SCONFIG	0xFFDA1E00	0xFFDA1FDF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

dma_SCOMPID Address Map

This address space holds peripheral ID information.

Module Instance	Base Address	End Address
i_dma_SCOMPID	0xFFDA1FE0	0xFFDA1FFF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Document Revision History

Table 16-7: Document Revision History

Date	Version	Changes
October 2016	2016.10.28	Maintenance release
May 2016	2016.05.27	Maintenance release
May 2016	2016.05.03	Maintenance release
November 2015	2015.11.02	• Updated link point to the HPS Address Map and Register Definitions • Added information about the instruction fetch cache properties
May 2015	2015.05.04	• Added Synopsys handshake rules.
December 2014	2014.12.15	Maintenance release
August 2014	2014.08.18	Initial release

Ethernet Media Access Controller 17

2016.10.28

a10_5v4

The hard processor system (HPS) provides three Ethernet media access controller (EMAC) peripherals. Each EMAC can be used to transmit and receive data at 10/100/1000 Mbps over Ethernet connections in compliance with the IEEE 802.3 specification. The EMACs are instances of the Synopsys DesignWare Universal 10/100/1000 Ethernet MAC (version 3.72a).

The EMAC has an extensive memory-mapped control and status register (CSR) set, which can be accessed by the ARM Cortex-A9® MPCore.

For an understanding of this chapter, you should be familiar with the basics of IEEE 802.3 media access control (MAC). (50)

Related Information

IEEE Standards Association

For complete information about IEEE 802.3 MAC, refer to the *IEEE 802.3 2008 Part 3: Carrier sense multiple access with Collision Detection (CSMA/CD) Access Method and Physical Layer Specifications*, available on the IEEE Standards Association website.

(50) Portions © 2016 Synopsys, Inc. Used with permission. All rights reserved. Synopsys & DesignWare are registered trademarks of Synopsys, Inc. All documentation is provided "as is" and without any warranty. Synopsys expressly disclaims any and all warranties, express, implied, or otherwise, including the implied warranties of merchantability, fitness for a particular purpose, and non-infringement, and any warranties arising out of a course of dealing or usage of trade.

†Paragraphs marked with the dagger (†) symbol are Synopsys Proprietary. Used with permission.

Features of the Ethernet MAC

MAC

- IEEE 802.3-2008 compliant
- Data rates of 10/100/1000 Mbps
- Full duplex and half duplex modes
  - IEEE 802.3x flow control automatic transmission of zero-quanta pause frame on flow control input deassertion
  - Optional forwarding of received pause control frames to the user
  - Packet bursting and frame extension in 1000 Mbps half-duplex
  - IEEE 802.3x flow control in full-duplex
  - Back-pressure support for half-duplex
- 4 KB TX FIFO RAM and 16 KB RX FIFO RAM with ECC support
- IEEE 1588-2002 and IEEE 1588-2008 precision networked clock synchronization
- IEEE 802.3-az, version D2.0 for Energy Efficient Ethernet (EEE)
- IEEE 802.1Q Virtual Local Area Network (VLAN) tag detection for reception frames
- Preamble and start-of-frame data (SFD) insertion in transmit and deletion in receive paths
- Automatic cyclic redundancy check (CRC) and pad generation controllable on a per-frame basis
- Options for automatic pad/CRC stripping on receive frames
- Programmable frame length supporting standard and jumbo Ethernet frames (with sizes up to 3800 bytes)
- Programmable inter-frame gap (IFG), from 40- to 96-bit times in steps of eight bits
- Preamble lengths of one, three, five and seven bytes supported
- Supports internal loopback asynchronous FIFO on the GMII/MII for debugging
- Supports a variety of flexible address filtering modes
  - Up to 31 additional 48-bit perfect destination address (DA) filters with masks for each byte
  - Up to 31 48-bit source address (SA) comparison check with masks for each byte
  - 256-bit hash filter (optional) for multicast and unicast DAs
  - Option to pass all multicast addressed frames
  - Promiscuous mode support to pass all frames without any filtering for network monitoring
  - Passes all incoming packets (as per filter) with a status report

DMA

- 32-bit interface
- Programmable burst size for optimal bus utilization
- Single-channel mode transmit and receive engines
- Byte-aligned addressing mode for data buffer support
- Dual-buffer (ring) or linked-list (chained) descriptor chaining
- Descriptors can each transfer up to 8 KB of data
- Independent DMA arbitration for transmit and receive with fixed priority or round robin

Management Interface

- 32-bit host interface to CSR set
- Comprehensive status reporting for normal operation and transfers with errors
- Configurable interrupt options for different operational conditions
- Per-frame transmit/receive complete interrupt control
- Separate status returned for transmission and reception packets

Acceleration

- Transmit and receive checksum offload for transmission control protocol (TCP), user datagram protocol (UDP), or Internet control message protocol (ICMP) over Internet protocol (IP)

PHY Interface

Different external PHY interfaces are provided depending on whether the Ethernet Controller signals are routed through the HPS I/O pins or the FPGA I/O pins.

The PHY interfaces supported using the HPS I/O pins are:

- Reduced Media Independent Interface (RMII)
- Reduced Gigabit Media Independent Interface (RGMII)

The PHY interfaces supported using the FPGA I/O pins are:

- Media Independent Interface (MII)
- Gigabit Media Independent Interface (GMII)
- Reduced Media Independent Interface (RMII) with additional required adaptor logic

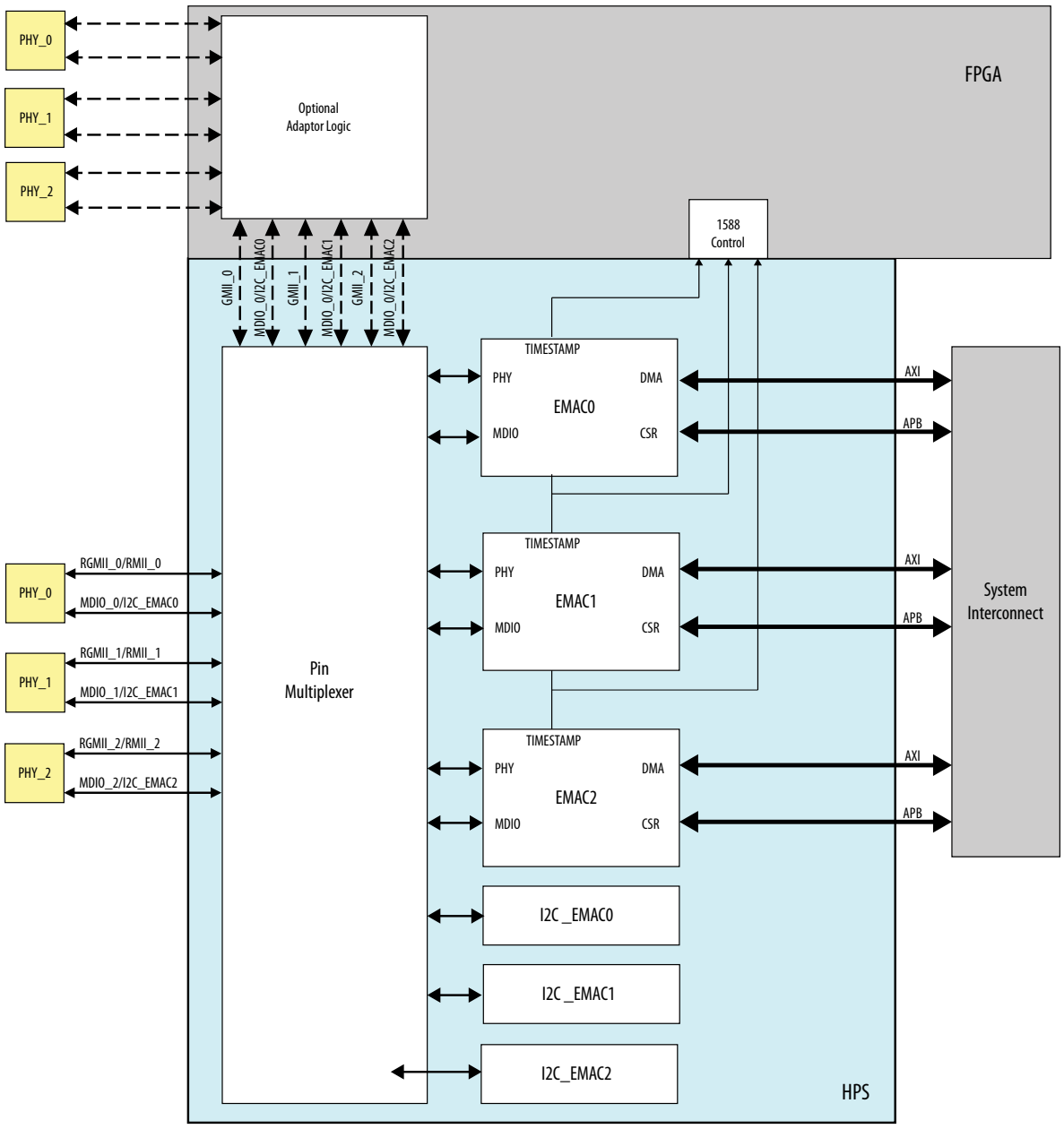
  Note: Additional adaptor logic for RMII not provided.
- Reduced Gigabit Media Independent Interface (RGMII) with additional required adaptor logic
- Serial Gigabit Media Independent Interface (SGMII) supported through transceiver I/O or high-speed low-voltage differential signaling (LVDS) with soft clock data recover (CDR) I/O with additional required adaptor logic

The Ethernet Controller has two choices for the management control interface used for configuration and status monitoring of the PHY:

- Management Data Input/Output (MDIO)
- I^2C PHY management through a separate I^2C module within the HPS

EMAC Block Diagram and System Integration

Figure 17-1: EMAC System Integration



EMAC Overview

Each EMAC is an internal bus master that sends Ethernet packets to and from the System Interconnect. The EMAC uses a descriptor ring protocol, where the descriptor contains an address to a buffer to fetch or store the packet data.

Each EMAC has an MDIO Management port to send commands to the external PHY. Alternatively, you can use an I^2C module in the HPS for the management interface.

Each EMAC has an IEEE 1588 Timestamp interface with 10 ns resolution. The ARM Cortex-A9 microprocessor unit (MPU) subsystem can use it to maintain synchronization between the time counters that are internal to the three MACs. The clock reference for the timestamp can be provided by the Clock Manager (`emac_ptp_clk`) or the FPGA fabric (`f2s_emac_ptp_ref_clk`). The clock reference is selected by the `ptp_clk_sel` bit in the `emac_global` register in the system manager.

Note: All three EMACs must use the same clock reference. In addition, EMAC0 can be configured to provide the timestamp for EMAC1, EMAC2, or both by setting the `ptp_ref_sel` bit in the `emac*` register in the System Manager.

EMAC Signal Description

The EMAC provides a variety of PHY interfaces and control options through the HPS and the FPGA I/Os.

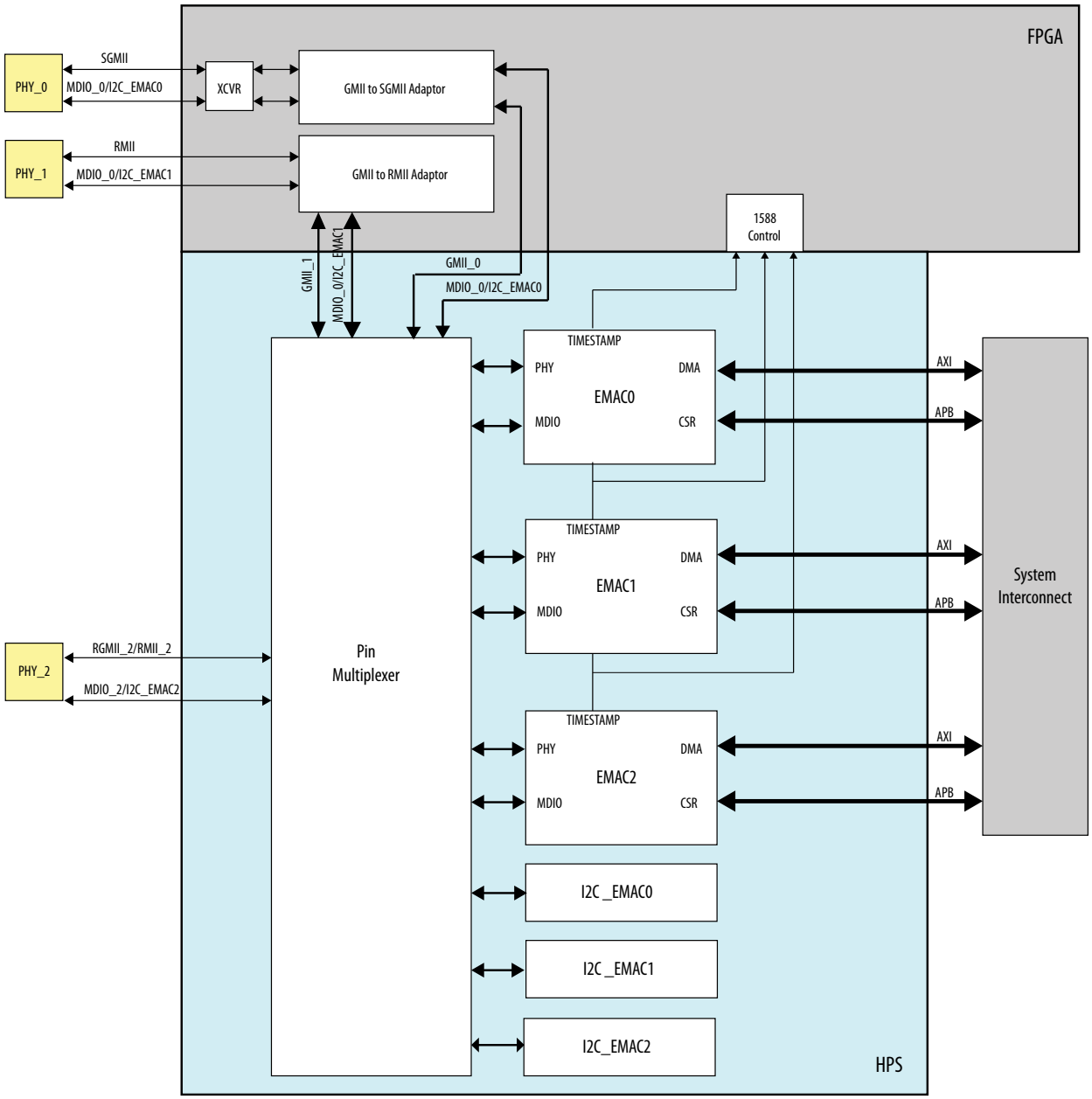
For designs in which the HPS is pin-limited, the EMAC signals can be routed through the FPGA as any one of the following interfaces by using soft adaptor logic in the FPGA:

- RMII/RGMII
- MII/GMII
- SGMII

The figure below depicts a design which routes the EMAC0 and EMAC1 signals through the FPGA to provide an RMII and SGMII interface. EMAC2 is routed through the HPS.

Refer to the "EMAC FPGA Interface Initialization" section to find out more information about configuring EMAC interfaces through FPGA.

Figure 17-2: EMAC to FPGA Routing Example



Related Information

EMAC FPGA Interface Initialization on page 17-71
Information on how to initialize GMII/MII interface

HPS EMAC I/O Signals

There are three EMACs available in the HPS. The following table lists the EMAC signals that can be routed from the EMACs to the HPS I/O pins. These signals provide the RMII/RGMII interface. For more

information on routing EMAC signals to the FPGA and the HPS I/O, refer to the *HPS Component Interfaces* chapter.

Note: The "n" in EMACn stands for the EMAC peripheral number.

Table 17-1: HPS EMAC I/O Signals

EMAC HPS I/O		In/Out	Width	Description
EMAC0_TX_CLK EMAC1_TX_CLK EMAC2_TX_CLK	Transmit Clock routed from one of three Qsys port signals emac[2:0]_phy_txclk_o	Out	1	This signal provides the transmit clock for RGMII (125/25/2.5 MHz in 1G/100M/10Mbps). This signal is one option for the common transmit and receive clock in RMII mode (50 MHz in 100M or 10 Mbps mode). The other possible source for the common transmit and receive clock is an external clock source, in which case EMACn_TX_CLK is left unconnected. In RMII mode, if this signal is the clock source for the receiver, then connect EMACn_TX_CLK to EMACn_RX_CLK. All PHY transmit signals generated by the EMAC are synchronous to this clock.
EMAC0_TXD[3:0] EMAC1_TXD[3:0] EMAC2_TXD[3:0]	PHY Transmit Data, routed from one of three groups of Qsys port signals emac[2:0]_phy_txd_o[3:0]	Out	4	This group of transmit data signals is driven by the MAC. Bits [3:0] provide the RGMII transmit data, and bits [1:0] provide the RMII transmit data. In RGMII mode, the data bus carries transmit data at double rate and are sampled on both the rising and falling edges of the transmit clock. The validity of the data is qualified with EMACn_TX_CTL.
EMAC0_TX_CTL EMAC1_TX_CTL EMAC2_TX_CTL	PHY Transmit Data Enable, routed from one of three Qsys port signals emac[2:0]_phy_txen	Out	1	This signal is driven by the EMAC component. In RGMII mode, this signal acts as the control signal for the transmit data, and is driven on both edges of the transmit clock, EMACn_TX_CLK. In RMII mode, this signal is high to indicate valid data.

EMAC HPS I/O		In/Out	Width	Description
EMAC0_RX_CLK EMAC1_RX_CLK EMAC2_RX_CLK	Receive Clock, routed to one of three Qsys port signals emac[2:0]_clk_rx_i	In	1	In RGMII mode, this clock frequency is 125/25/2.5 MHz in 1 G/100 M/10 Mbps modes. It is provided by the external PHY. All PHY signals received by the EMAC are synchronous to this clock. In RMII mode, this clock frequency is 50 MHz. The source of this clock can be: • An external source: In this case EMACn_TX_CLK should be left unconnected. • EMACn_TX_CLK: In this case, EMACn_TX_CLK must be connected to EMACn_RX_CLK.
EMAC0_RXD[3:0] EMAC1_RXD[3:0] EMAC2_RXD[3:0]	PHY Receive Data, routed to one of three groups of Qsys port signals emac[2:0]_phy_rxd[3:0]	In	4	These data signals are received from the PHY. In RGMII mode, data is received at double data rate with bits[3:0] valid on the rising and falling edges of EMACn_RX_CLK. In RMII mode, data is received at single data rate with bits [1:0] valid on the rising edge of EMACn_RX_CLK. The validity of the data is qualified with EMACn_RX_CTL.
EMAC0_RX_CTL EMAC1_RX_CTL EMAC2_RX_CTL	PHY Receive Data Valid, routed to one of three groups of Qsys port signals emac[2:0]_phy_rxdv.	In	1	This signal is driven by the PHY and functions as the receive control signal used to qualify the data received on EMACn_RXD[3:0]. This signal is sampled on both edges of the clock in RGMII mode. See row above for clock to data relationships across the modes.

Related Information

EMAC HPS Interface Initialization on page 17-72
Information on how to initialize RGMII/RMII interface

FPGA EMAC I/O Signals

Three Ethernet Media Access Controllers are provided in the HPS. The table below describes the signals that are available from each Ethernet Media Access Controller to the FPGA I/O. For more information on routing EMAC signals to the FPGA and HPS I/O, refer to the *HPS Component Interfaces* chapter. See the HPS I/O table for general clock to data relationships across the modes.

Table 17-2: FPGA EMAC I/O Signals

Signal Name		In/Out	Width	Description
emac_clk_tx_i	Transmit Clock	In	1	This is the transmit clock (2.5 MHz/25 MHz) provided by the MII PHYs only. This clock comes from the FPGA Interface and is used for TX data capture. This clock is not used in GMII mode. **Note:** This clock must be able to perform glitch free switching between 2.5 and 25 MHz.
emac_phy_txclk_o	Transmit Clock Output	Out	1	In GMII mode, this signal is the transmit clock output to the PHY to sample data. For MII, this clock is unused.
emac_phy_txd_o[7:0]	PHY Transmit Data	Out	8	These are a group of eight transmit data signals driven by the EMAC. All eight bits provide the GMII transmit data byte. For the lower speed MII operation, only bits[3:0] are used. The validity of the data is qualified with phy_txen_o and phy_txer_o. Synchronous to phy_txclk_o.
emac_phy_txen_o	PHY Transmit Data Enable	Out	1	This signal is driven by the EMAC and is used in GMII mode. When driven high, this signal indicates that valid data is being transmitted on the clk_tx_o bus.
emac_phy_txer_o	PHY Transmit Error	Out	1	This signal is driven by the EMAC and when high, indicates a transmit error or carrier extension on the phy_txd bus. It is also used to signal low power states in Energy Efficient Ethernet operation.

Signal Name		In/Out	Width	Description
emac_rst_clk_tx_n_o	Transmit Clock Reset output	Out	1	Transmit clock reset output to the FPGA fabric, which is the internal synchronized reset to clk_tx_int output from the EMAC. May be used by logic implemented in the FPGA fabric as desired. The reset pulse width of the rst_clk_tx_n_o signal is three transmit clock cycles.
emac_clk_rx_i	Receive Clock	In	1	Receive clock from external PHY. For GMII, the clock frequency is 125 MHz. For MII, the receive clock is 25 MHz for 100 Mbps and 2.5 MHz for 10 Mbps.
emac_phy_rxd_i[7:0]	PHY Receive Data	In	8	This is an eight-bit receive data bus from the PHY. In GMII mode, all eight bits are sampled. The validity of the data is qualified with phy_rxdv_i and phy_rxer_i. For lower speed MII operation, only bits [3:0] are sampled. These signals are synchronous to phy_clk_rx_i.
emac_phy_rxdv_i	PHY Receive Data Valid	In	1	This signal is driven by PHY. In GMII mode, when driven high, it indicates that the data on the phy_rxd bus is valid. It remains asserted continuously from the first recovered byte of the frame through the final recovered byte.
emac_phy_rxer_i	PHY Receive Error	In	1	This signal indicates an error or carrier extension (GMII) in the received frame. This signal is synchronous to phy_clk_rx_i.
emac_rst_clk_rx_n_o	Receive clock reset output.	Out	1	Receive clock reset output. The reset pulse width of the rst_clk_rx_n_o signal is three transmit clock cycles.

Signal Name		In/Out	Width	Description
emac_phy_mac_speed_o[1:0]	EMAC Speed	Out	2	This signal indicates the 10-Mbps, 100-Mbps or 1000-Mbps operating speed and reflects the Port Select and FES bit of the MAC Configuration register. The signal value encodings are: • 0x0: 1000 Mbps (GMII) • 0x1: 1000 Mbps (GMII) • 0x2: 10 Mbps (MII) • 0x3: 100 Mbps (MII)
emac_phy_crs_i	PHY Carrier Sense	In	1	This signal is asserted by the PHY when either the transmit or receive medium is not idle. The PHY de-asserts this signal when both transmit and receive interfaces are idle. This signal is not synchronous to any clock.
emac_phy_col_i	PHY Collision Detect	In	1	This signal, valid only when operating in half duplex, is asserted by the PHY when a collision is detected on the medium. This signal is not synchronous to any clock.

Related Information

EMAC FPGA Interface Initialization on page 17-71
Information on how to initialize GMII/MII interface

PHY Management Interface

The HPS can provide support for either MDIO or I^2C PHY management interfaces.

MDIO Interface

The MDIO interface signals are synchronous to l4_mp_clk in all supported modes.

Note: The MDIO interface signals can be routed to both the FPGA and HPS I/O.

Table 17-3: PHY MDIO Management Interface

Signal	HPS I/O Pin Name	In/Out	Width	Description
emac_gmii_mdi_i	EMACn_MDIO	In	1	Management Data In. The PHY generates this signal to transfer register data during a read operation. This signal is driven synchronously with the gmii_mdc_o clock.
emac_gmii_mdo_o		Out	1	Management Data Out. The EMAC uses this signal to transfer control and data information to the PHY.
emac_gmii_mdo_o_e		Out	1	Management Data Output Enable. This enable signal drives the gmii_mdo_o signal from an external three-state I/O buffer. This signal is asserted whenever valid data is driven on the gmii_mdo_o signal. The active state of this signal is high.
emac_gmii_mdc_o	EMACn_MDC	Out	1	Management Data Clock. The EMAC provides timing reference for the gmii_mdi_i and gmii_mdo_o signals on MII through this aperiodic clock. The maximum frequency of this clock is 2.5 MHz. This clock is generated from the application clock through a clock divider.

I²C External PHY Management Interface

Some PHY devices use the I²C instead of MDIO for their control interface. Small form factor pluggable (SFP) optical or pluggable modules are often among those with this interface.

The HPS or FPGA can use three of the five general purpose I²C peripherals for controlling the PHY devices:

- i2c_emac_0 at base address 0xFFC02400
- i2c_emac_1 at base address 0xFFC02500
- i2c_emac_2 at base address 0xFFC02600

EMAC Internal Interfaces

DMA Master Interface

The DMA interface acts as a bus master on the system interconnect. Two types of data are transferred on the interface: data descriptors and actual data packets. The interface is very efficient in transferring full duplex Ethernet packet traffic. Read and write data transfers from different DMA channels can be

performed simultaneously on this port, except for transmit descriptor reads and write-backs, which cannot happen simultaneously.

DMA transfers are split into a software configurable number of burst transactions on the interface. The `AXI_Bus_Mode` register in the `dmagrp` group is used to configure bursting behavior.

The interface assigns a unique ID for each DMA channel and also for each read DMA or write DMA request in a channel. Data transfers with distinct IDs can be reordered and interleaved.

The DMA interface can be configured to perform cacheable accesses. This configuration can be done in the System Manager when the DMA interface is inactive.

Write data transfers are generally performed as posted writes with OK responses returned as soon as the system interconnect has accepted the last beat of a data burst. Descriptors (status or timestamp), however, are always transferred as non-posted writes in order to prevent race conditions with the transfer complete interrupt logic.

The slave may issue an error response. When that happens, the EMAC disables the DMA channel that generated the original request and asserts an interrupt signal. The host must reset the EMAC with a hard or soft reset to restart the DMA to recover from this condition.

The EMAC supports up to 16 outstanding transactions on the interface. Buffering outstanding transactions smooths out back pressure behavior improving throughput when resource contention bottlenecks arise under high system load conditions.

Related Information

- **DMA Controller** on page 17-18
  Information regarding DMA Controller functionality
- **System Manager** on page 5-1

Timestamp Interface

The timestamp clock reference can come from either the Clock Manager or the FPGA fabric. If the FPGA has implemented the serial capturing of the timestamp interface, then the FPGA must provide the PTP clock reference.

Each EMAC provides its internal timestamp as an output. In some applications, it is advantageous to allow the FPGA fabric access to the Ethernet timestamp. In that case, the timestamp output from each EMAC is sampled in the `clk_ptp_ref_i` clock domain and serially shifted out to the FPGA fabric. The PTP timestamp clock must be selected to come from the FPGA fabric if the serial timestamp is used in the FPGA.

In addition to providing a timestamp clock reference, the FPGA can monitor the pulse-per-second output from each EMAC module and trigger a snapshot from each auxiliary time stamp timer.

The following table lists the EMAC to FPGA IEEE1588 Timestamp Interface signals to and from each EMAC module.

Table 17-4: EMAC to FPGA IEEE 1588 Timestamp Interface Signals

Signal Name		In/Out	Width	Description
f2s_emac_ptp_ref_clk	Timestamp PTP Clock reference from the FPGA	In	1	Used as PTP Clock reference for each EMAC when the FPGA has implemented Timestamp capture interface. The timestamp clock is common to all three EMACs. The frequency of this clock can be up to 100 MHz.
ptp_tstmp_en	Timestamp Serial Interface Enable	Out	1	When the local timestamp of each EMAC is sampled, the enable signal is pulsed with the first of 64 bits of serially shifted data. Synchronous to f2s_emac_ptp_ref_clk.
ptp_tstmp_data	Timestamp Serial Interface Data	Out	1	The 64-bit sampled timestamp is shifted serially to the FPGA fabric from the EMAC. The enable is asserted only on the first bit. The first bit transferred is the least significant bit of the sampled ptp_timestamp[63:0], or ptp_timestamp[0].
ptp_pps_o	Pulse Per Second Output	Out	1	This signal is asserted based on the PPS mode selected in the Register 459 (PPS Control Register). Otherwise, this pulse signal is asserted every time the seconds counter is incremented. This signal is synchronous to f2s_emac_ptp_ref_clk and may only be sampled if the FPGA clock is used as timestamp reference.

Signal Name		In/Out	Width	Description
ptp_aux_ts_trig_i	Auxiliary Timestamp Trigger	In	1	This signal is asserted to take an auxiliary snapshot of the time. The rising edge of this internal signal is used to trigger the auxiliary snapshot. The signal is synchronized internally with clk_ptp_ref_i which results in an additional delay of 3 cycles. This input is asynchronous input and its assertion period must be greater than 2 PTP active clocks to be sampled.

Each EMAC supports either internal or external timestamp reference. In addition, EMAC0 has the option to be the master that provides the timestamp to EMAC1 and EMAC2. In this configuration, EMAC0 must be programmed to select internal timestamp generation in the System Manager and EMAC1 and EMAC2 must be programmed to select external timestamp generation.

Related Information

IEEE 1588-2002 Timestamps on page 17-48
More information regarding IEEE 1588-2002 timestamp functionality

System Manager Configuration Interface

The System Manager configures several static EMAC functions as shown in the following table. Software must configure these functions appropriately prior to using the EMAC module. Refer to the *Ethernet Programming Model* section for more details regarding pertinent System Manager registers.

Table 17-5: System Manager Control Settings

Function	Description
PHY Select	Select RESET, RGMII, RMII or GMII/MII as the PHY interface. The RESET mode is the default out of reset and configures the EMAC to use an internal clock rather than depending on a PHY to provide and active clock. The RESET mode cannot be used with any PHY, and another setting must be programmed before attempting to communicate with a PHY.
PTP Timestamp Reference Select	This field selects if the Timestamp reference is internally or externally generated. EMAC0 must be set to Internal Timestamp. EMAC0 may be the master to generate the timestamp for EMAC1 and EMAC2. EMAC1 and EMAC2 may be set to either Internal or External.

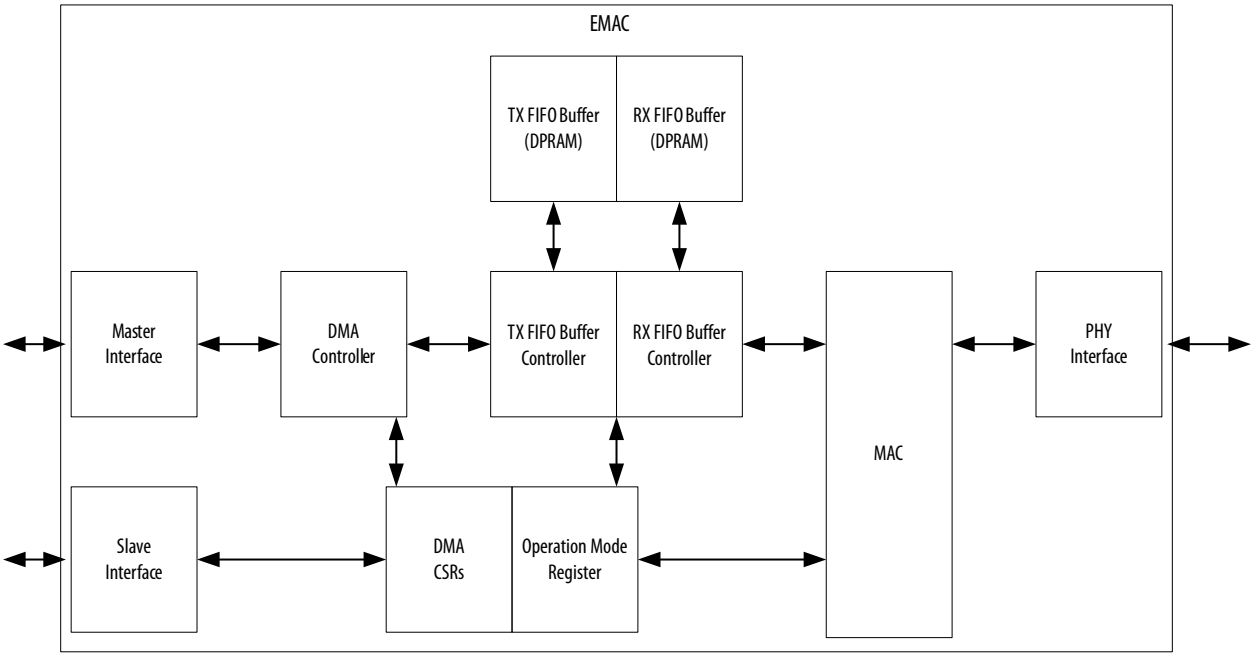
Function	Description
PTP Timestamp Clock Select	Selects the source of the PTP reference clock between `emac_ptp_clk` from the Clock Manager or `f2s_emac_ptp_ref_clk` from the FPGA Fabric. All three EMAC modules must use the same reference clock.
AXI Cache and Protection Settings	Static settings are provided to drive the ARCACHE, AWCACHE, ARPROT, and AWPROT signals for the AXI DMA bus.
FPGA Interface Enable	This field enables logic from the FPGA. This signal is only for safety to prevent spurious inputs from the FPGA before the FPGA is configured.

Related Information

- **System Level EMAC Configuration Registers** on page 17-69
  Information regarding system level registers involved in EMAC initialization
- **System Manager** on page 5-1

Functional Description of the EMAC

Figure 17-3: EMAC High-Level Block Diagram with Interfaces



There are two host interfaces to the Ethernet MAC. The management host interface, a 32-bit slave interface, provides access to the CSR set regardless of whether or not the EMACs are used directly through the FPGA fabric. The data interface is a 32-bit master interface, and it controls data transfer between the direct memory access (DMA) controller channels and the rest of the HPS system through the L3 interconnect.

The built-in DMA controller is optimized for data transfer between the MAC controller and system memory. The DMA controller has independent transmit and receive engines and a CSR set. The transmit engine transfers data from system memory to the device port, while the receive engine transfers data from the device port to the system memory. The controller uses descriptors to efficiently move data from source to destination with minimal host intervention.

The EMAC also contains FIFO buffer memory to buffer and regulate the Ethernet frames between the application system memory and the EMAC module. Each EMAC module has one 4 KB TX FIFO and one 16 KB RX FIFO. On transmit, the Ethernet frames write into the transmit FIFO buffer, and eventually trigger the EMAC to perform the transfer. Received Ethernet frames are stored in the receive FIFO buffer and the FIFO buffer fill level is communicated to the DMA controller. The DMA controller then initiates the configured burst transfers. Receive and transmit transfer statuses are read by the EMAC and transferred to the DMA.

Transmit and Receive Data FIFO Buffers

Each EMAC component has associated transmit and receive data FIFO buffers to regulate the frames between the application system memory and the EMAC. The RX FIFO buffer is a 16 KB dual-ported memory and the TX FIFO buffer is a 4 KB dual-ported memory. Both buffers are designed to support jumbo frames. A FIFO buffer word consists of:

- Data: 32 bits
- Sideband:
  - Byte enables: 2 bits
  - End of frame (EOF): 1 bit
  - Error correction code (ECC): 7 bits

The FIFO RAMs are each supported by an ECC controller that performs single-bit error detection and correction and double-bit error detection. The ECC Controllers have a dedicated hardware block for memory data initialization and can log error events and generate interrupts on single and double-error events. See the *Error Checking and Correction (ECC) Controller* for more information regarding the function of the ECC RAMs.

TX FIFO

The time at which data is sent from the TX FIFO to the EMAC is dependent on the transfer mode selected:

- Cut-through mode: Data is popped from the TX FIFO when the number of bytes in the TX FIFO crosses the configured threshold level (or when the end of the frame is written before the threshold is crossed). The threshold level is configured using the TTC bit of Register 0 (Bus Mode Register).

  Note: After more than 96 bytes (or 548 bytes in 1000 Mbps mode) are popped to the EMAC, the TX FIFO controller frees that space and makes it available to the DMA and a retry is not possible.
- Store-and-Forward mode: Data is popped from the TX FIFO when one or more of the following conditions are true:
  - A complete frame is stored in the FIFO
  - The TX FIFO becomes almost full

The application can flush the TX FIFO of all contents by setting bit 20 (FTF) of Register 6 (Operation Mode Register). This bit is self-clearing and initializes the FIFO pointers to the default state. If the FTF bit is set during a frame transfer to the EMAC, further transfers are stopped because the FIFO is considered empty. This cessation causes an underflow event and a runt frame to be transmitted and the corresponding status word is forwarded to the DMA.

If a collision occurs in half-duplex mode operation before an end of the frame, a retry attempt is sent before the end of the frame is transferred. When notified of the retransmission, the MAC pops the frame from the FIFO again.

Note: Only packets of 3800 bytes or less can be supported when the checksum offload feature is enabled by software.

RX FIFO

Frames received by the EMAC are pushed into the RX FIFO. The fill level of the RX FIFO is indicated to the DMA when it crosses the configured receive threshold which is programmed by the RTC field of Register 6 (Operation Mode Register). The time at which data is sent from the RX FIFO to the DMA is dependent on the configuration of the RX FIFO:

- Cut-through (default) mode: When 64 bytes or a full packet of data is received into the FIFO, data is popped out of the FIFO and sent to the DMA until a complete packet has been transferred. Upon completion of the end-of-frame transfer, the status word is popped and sent to the DMA.
- Store and forward mode: A frame is read out only after being written completely in the RX FIFO. This mode is configured by setting the RSF bit of Register 6 (Operation Mode Register).

If the RX FIFO is full before it receives the EOF data from the EMAC, an overflow is declared and the whole frame (including the status word) is dropped and the overflow counter in the DMA, (Register 8) Missed Frame and Buffer Overflow Counter Register, is incremented. This outcome is true even if the Forward Error Frame (FEF) bit of Register 6 (Operation Mode Register) is set. If the start address of such a frame has already been transferred, the rest of the frame is dropped and a dummy EOF is written to the FIFO along with the status word. The status indicates a partial frame because of overflow. In such frames, the Frame Length field is invalid. If the RX FIFO is configured to operate in the store-and-forward mode and if the length of the received frame is more than the FIFO size, overflow occurs and all such frames are dropped.

Note: In store-and-forward mode, only received frames with length 3800 bytes or less prevent overflow errors and frames from being dropped.

Related Information

Error Checking and Correction Controller on page 11-1
Error Checking and Correction (ECC) controllers provide single- and double-bit error memory protection for integrated on-chip RAM and peripheral RAMs within the hard processor system (HPS).

DMA Controller

The DMA has independent transmit and receive engines, and a CSR space. The transmit engine transfers data from system memory to the device port or MAC transaction layer (MTL), while the receive engine transfers data from the device port to the system memory. Descriptors are used to efficiently move data from source to destination with minimal Host CPU intervention. The DMA is designed for packet-oriented data transfers such as frames in Ethernet. The controller can be programmed to interrupt the Host CPU for situations such as frame transmit and receive transfer completion as well as error conditions.

The DMA and the Host driver communicate through two data structures:†

- Control and Status registers (CSR)†
- Descriptor lists and data buffers †

Descriptor Lists and Data Buffers†

The DMA transfers data frames received by the MAC to the receive Buffer in the Host memory, and transmit data frames from the transmit Buffer in the Host memory. Descriptors that reside in the Host memory act as pointers to these buffers.†

There are two descriptor lists: one for reception and one for transmission. The base address of each list is written into Register 3 (Receive Descriptor List Address Register) and Register 4 (Transmit Descriptor List Address Register), respectively. A descriptor list is forward linked (either implicitly or explicitly). The last descriptor may point back to the first entry to create a ring structure. Explicit chaining of descriptors is accomplished by setting the second address chained in both receive and transmit descriptors (RDES1[14] and TDES0[20]). The descriptor lists resides in the Host physical memory address space. Each descriptor can point to a maximum of two buffers. This enables two buffers to be used, physically addressed, rather than contiguous buffers in memory.†

A data buffer resides in the Host physical memory space, and consists of an entire frame or part of a frame, but cannot exceed a single frame. Buffers contain only data, buffer status is maintained in the descriptor. Data chaining refers to frames that span multiple data buffers. However, a single descriptor cannot span multiple frames. The DMA skips to the next frame buffer when end-of-frame is detected. Data chaining can be enabled or disabled.†

Figure 17-4: Descriptor Ring Structure

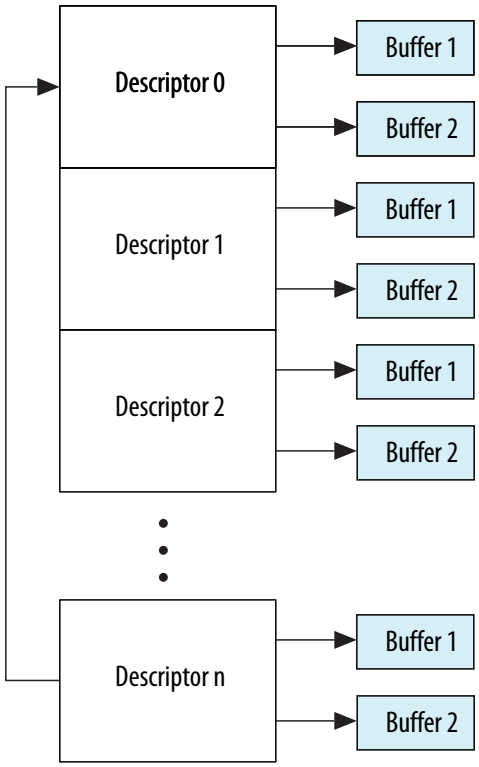
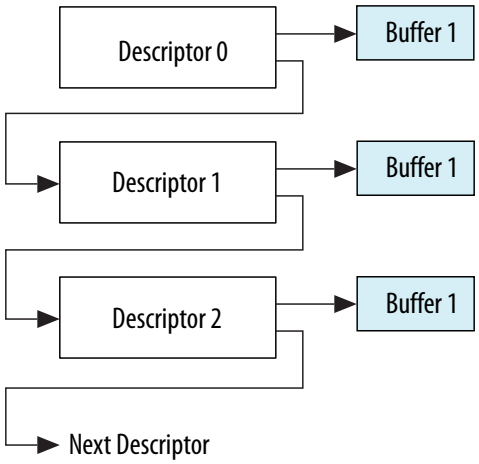


Figure 17-5: Descriptor Chain Structure



Note: You can select a descriptor structure during RTL configuration. The control bits in the descriptor structure are assigned so that the application can use an 8 KB buffer. All descriptions refer to the default descriptor structure.

Related Information

Ethernet MAC Address Map and Register Definitions on page 17-78
Information about control and status registers

Host Bus Burst Access

The DMA attempts to execute fixed-length burst transfers on the master interface if configured to do so through the FB bit of Register 0 (Bus Mode Register). The maximum burst length is indicated and limited by the PBL field (Bits [13:8]) Register 0 (Bus Mode Register). The receive and transmit descriptors are always accessed in the maximum possible (limited by packet burst length (PBL) or 16 * 8/bus width) burst size for the 16- bytes to be read.

The transmit DMA initiates a data transfer only when the MTL transmit FIFO has sufficient space to accommodate the configured burst or the number of bytes remaining in the frame (when it is less than the configured burst length). The DMA indicates the start address and the number of transfers required to the master interface. When the interface is configured for fixed-length burst, it transfers data using the best combination of INCR4, 8, or 16 and SINGLE transactions. When not configured for fixed-length burst, it transfers data using INCR (undefined length) and SINGLE transactions.

The receive DMA initiates a data transfer only when sufficient data to accommodate the configured burst is available in MTL receive FIFO buffer or when the end of frame (when it is less than the configured burst length) is detected in the receive FIFO buffer. The DMA indicates the start address and the number of transfers required to the master interface. When the interface is configured for fixed-length burst, it transfers data using the best combination of INCR4, 8, or 16 and SINGLE transactions. If the end-of-frame is reached before the fixed burst ends on the interface, then dummy transfers are performed in order to complete the fixed burst. If the FB bit of Register 0 (Bus Mode Register) is clear, it transfers data using INCR (undefined length) and SINGLE transactions.

When the interface is configured for address aligned words, both DMA engines ensure that the first burst transfer initiated is less than or equal to the size of the configured packet burst length. Thus, all subsequent

beats start at an address that is aligned to the configured packet burst length. The DMA can only align the address for beats up to size 16 (for PBL > 16), because the interface does not support more than INCR16.

Host Data Buffer Alignment

The transmit and receive data buffers do not have any restrictions on start address alignment. For example, in systems with 32-bit memory, the start address for the buffers can be aligned to any of the four bytes. However, the DMA always initiates transfers with address aligned to the bus width with dummy data for the byte lanes not required. This typically happens during the transfer of the beginning or end of an Ethernet frame. The software driver should discard the dummy bytes based on the start address of the buffer and size of the frame.†

Example: Buffer Read

If the transmit buffer address is 0x00000FF2, and 15 bytes must be transferred, then the DMA reads five full words from address 0x00000FF0, but when transferring data to the MTL transmit FIFO buffer, the extra bytes (the first two bytes) are dropped or ignored. Similarly, the last three bytes of the last transfer are also ignored. The DMA always ensures it transfers data in 32-bit increments to the MTL transmit FIFO buffer, unless it is the end-of-frame.

Example: Buffer Write

If the receive buffer address is 0x00000FF2 and 16 bytes of a received frame must be transferred, then the DMA writes 3 full words from address 0x00000FF0. But the first two bytes of first transfer and the last two bytes of the fourth transfer have dummy data.

Buffer Size Calculations

The DMA does not update the size fields in the transmit and receive descriptors. The DMA updates only the status fields (RDES and TDES) of the descriptors. The driver must perform the size calculations.

The transmit DMA transfers the exact number of bytes (indicated by the buffer size field of TDES1) to the MAC. If a descriptor is marked as th first (FS bit of TDES1 is set), then the DMA marks the first transfer from the buffer as the start of frame. If a descriptor is marked as the last (LS bit of TDES1), then the DMA marks the last transfer from that data buffer as the end-of-frame to the MTL.

The receive DMA transfers data to a buffer until the buffer is full or the end-of-frame is received from the MTL. If a descriptor is not marked as the last (LS bit of RDES0), then the descriptor's corresponding buffer(s) are full and the amount of valid data in a buffer is accurately indicated by its buffer size field minus the data buffer pointer offset when the FS bit of that descriptor is set. The offset is zero when the data buffer pointer is aligned to the data bus width. If a descriptor is marked as the last, then the buffer may not be full (as indicated by the buffer size in RDES1). To compute the amount of valid data in this final buffer, the driver must read the frame length (FL bits of RDES0[29:16]) and subtract the sum of the buffer sizes of the preceding buffers in this frame. The receive DMA always transfers the start of next frame with a new descriptor.

Note: Even when the start address of a receive buffer is not aligned to the data width of system bus, the system should allocate a receive buffer of a size aligned to the system bus width. For example, if the system allocates a 1,024-byte (1 KB) receive buffer starting from address 0x1000, the software can program the buffer start address in the receive descriptor to have a 0x1002 offset. The receive DMA writes the frame to this buffer with dummy data in the first two locations (0x1000 and 0x1001). The actual frame is written from location 0x1002. Thus, the actual useful space in this buffer is 1,022 bytes, even though the buffer size is programmed as 1,024 bytes, because of the start address offset.

Transmission

The DMA can transmit with or without an optional second frame (OSF).

Related Information

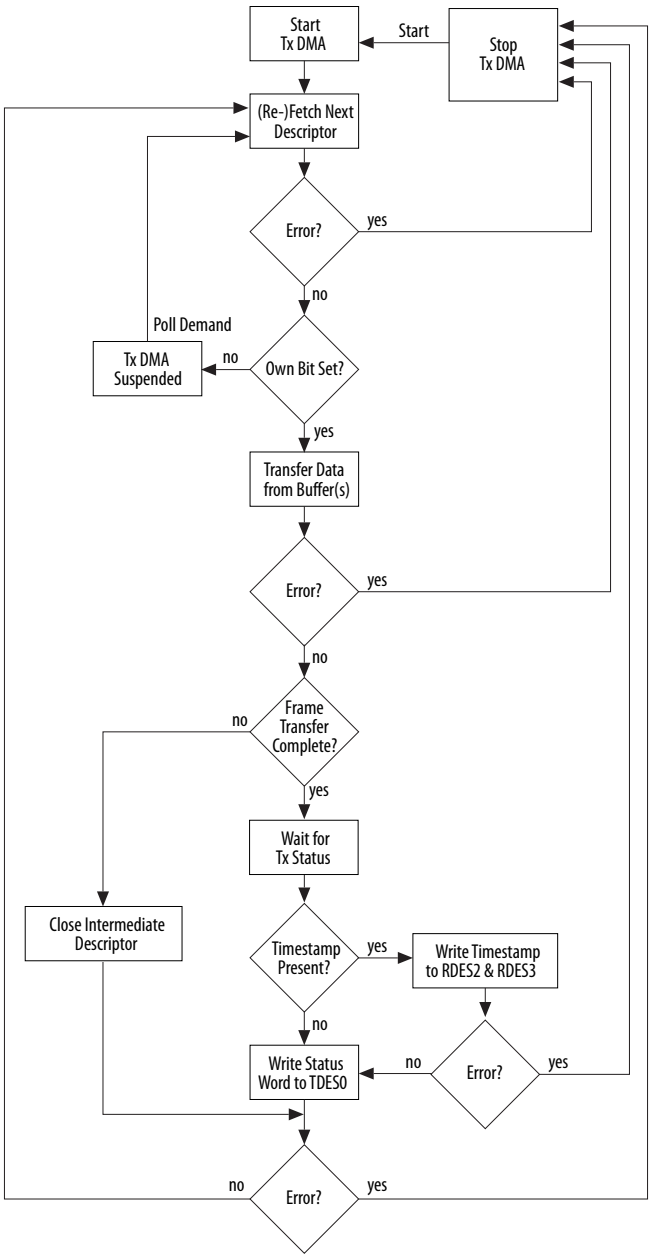
Transmit Descriptor on page 17-31

TX DMA Operation: Default (Non-OSF) Mode

The transmit DMA engine in default mode proceeds as follows: †

1. The Host sets up the transmit descriptor (TDES0-TDES3) and sets the Own bit (TDES0[31]) after setting up the corresponding data buffer(s) with Ethernet frame data. †
2. When Bit 13 (ST) of Register 6 (Operation Mode Register) is set, the DMA enters the Run state. †
3. While in the Run state, the DMA polls the transmit descriptor list for frames requiring transmission. After polling starts, it continues in either sequential descriptor ring order or chained order. If the DMA detects a descriptor flagged as owned by the Host (TDES0[31] = 0), or if an error condition occurs, transmission is suspended and both the Bit 2 (Transmit Buffer Unavailable) and Bit 16 (Normal Interrupt Summary) of the Register 5 (Status Register) are set. The transmit Engine proceeds to **step 9**.
4. If the acquired descriptor is flagged as owned by DMA (TDES0[31] = 1), the DMA decodes the transmit Data Buffer address from the acquired descriptor.
5. The DMA fetches the transmit data from the Host memory and transfers the data to the MTL for transmission. †
6. If an Ethernet frame is stored over data buffers in multiple descriptors, the DMA closes the intermediate descriptor and fetches the next descriptor. Repeat **step 3**, **step 4**, and **step 5** until the end-of-Ethernet-frame data is transferred to the MTL. †
7. When frame transmission is complete, if IEEE 1588 timestamping was enabled for the frame (as indicated in the transmit status) the timestamp value obtained from MTL is written to the transmit descriptor (TDES2 and TDES3) that contains the end-of-frame buffer. The status information is then written to this transmit descriptor (TDES0). Because the Own bit is cleared during this step, the Host now owns this descriptor. If timestamping was not enabled for this frame, the DMA does not alter the contents of TDES2 and TDES3. †
8. Bit 0 (Transmit Interrupt) of Register 5 (Status Register) is set after completing transmission of a frame that has Interrupt on Completion (TDES1[31]) set in its Last descriptor. The DMA engine then returns to **step 3**. †
9. In the Suspend state, the DMA tries to re-acquire the descriptor (and thereby return to **step 3**) when it receives a Transmit Poll demand and the Underflow Interrupt Status bit is cleared. †

Figure 17-6: TX DMA Operation in Default Mode



TX DMA Operation: OSF Mode

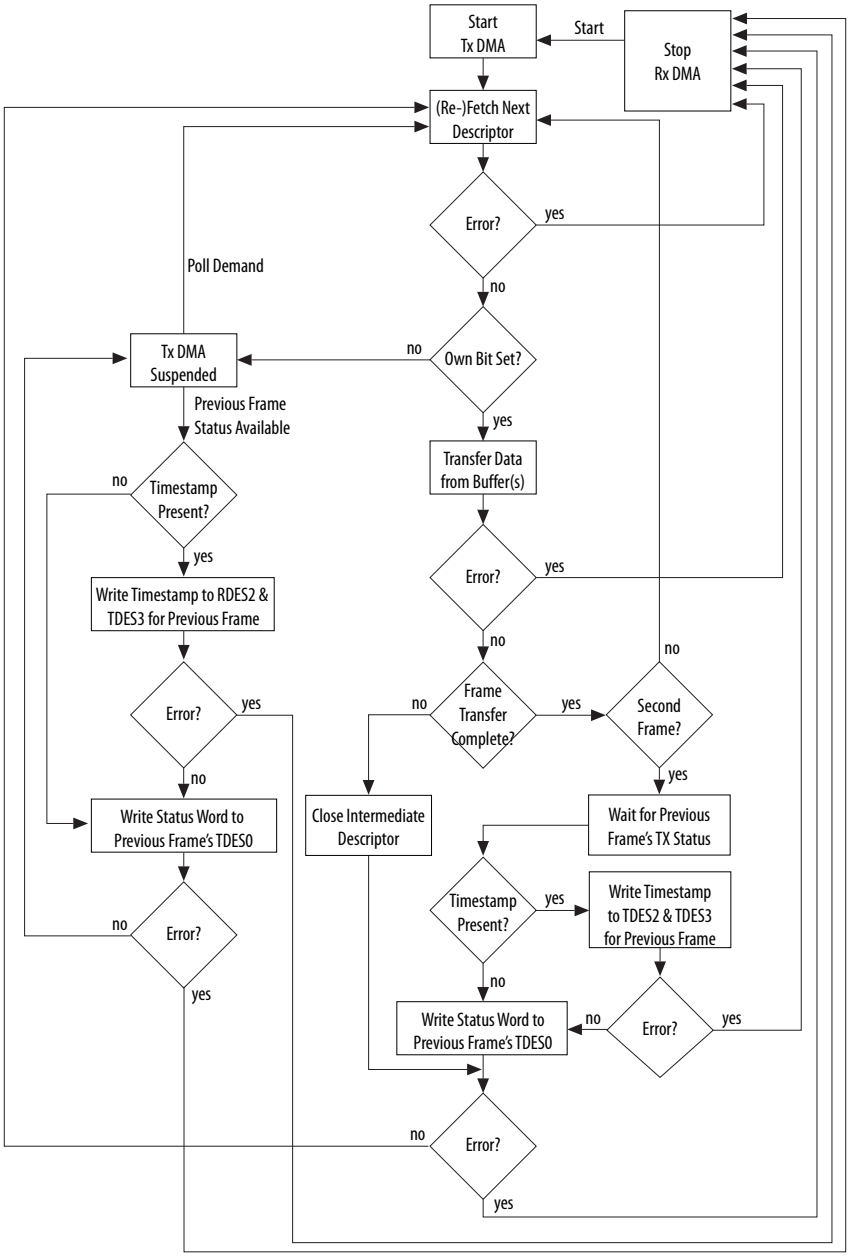
While in the Run state, the transmit process can simultaneously acquire two frames without closing the Status descriptor of the first [if Bit 2 (OSF) in Register 6 (Operation Mode Register) is set]. As the transmit process finishes transferring the first frame, it immediately polls the transmit descriptor list for the second frame. If the second frame is valid, the transmit process transfers this frame before writing the first frame's status information. †

In OSF mode, the Run state transmit DMA operates in the following sequence: †

1. The DMA operates as described in steps 1 - 6 of the **TX DMA Operation: Default (Non-OSF) Mode** on page 17-22 section.
2. Without closing the previous frame's last descriptor, the DMA fetches the next descriptor. †
3. If the DMA owns the acquired descriptor, the DMA decodes the transmit buffer address in this descriptor. If the DMA does not own the descriptor, the DMA goes into Suspend mode and skips to **step 7**.†
4. The DMA fetches the transmit frame from the Host memory and transfers the frame to the MTL until the End-of-frame data is transferred, closing the intermediate descriptors if this frame is split across multiple descriptors. †
5. The DMA waits for the previous frame's frame transmission status and timestamp. Once the status is available, the DMA writes the timestamp to TDES2 and TDES3, if such timestamp was captured (as indicated by a status bit). The DMA then writes the status, with a cleared Own bit, to the corresponding TDES0, thus closing the descriptor. If timestamping was not enabled for the previous frame, the DMA does not alter the contents of TDES2 and TDES3. †
6. If enabled, the transmit interrupt is set, the DMA fetches the next descriptor, then proceeds to **step 3** (when Status is normal). If the previous transmission status shows an underflow error, the DMA goes into Suspend mode (**step 7**). †
7. In Suspend mode, if a pending status and timestamp are received from the MTL, the DMA writes the timestamp (if enabled for the current frame) to TDES2 and TDES3, then writes the status to the corresponding TDES0. It then sets relevant interrupts and returns to Suspend mode. †
8. The DMA can exit Suspend mode and enter the Run state (go to **step 1** or **step 2** depending on pending status) only after receiving a Transmit Poll demand (Register 1 (Transmit Poll Demand Register). †

Note: As the DMA fetches the next descriptor in advance before closing the current descriptor, the descriptor chain should have more than two different descriptors for correct and proper operation. †

Figure 17-7: TX DMA Operation in OSF Mode



Transmit Frame Processing

The transmit DMA expects that the data buffers contain complete Ethernet frames, excluding preamble, pad bytes, and FCS fields and that the DA, SA, and Type/Len fields contain valid data. If the transmit descriptor indicates that the MAC must disable CRC or PAD insertion, the buffer must have complete Ethernet frames (excluding preamble), including the CRC bytes. †

Frames can be datachained and can span several buffers. Frames must be delimited by the First Descriptor (TDES1[29]) and the Last Descriptor (TDES1[30]), respectively. †

As transmission starts, the First Descriptor must have (TDES1[29]) set. When this occurs, frame data transfers from the Host buffer to the MTL transmit FIFO buffer. Concurrently, if the current frame has the Last Descriptor (TDES1[30]) clear, the transmit Process attempts to acquire the Next descriptor. The transmit Process expects this descriptor to have TDES1[29] clear. If TDES1[30] is clear, it indicates an intermediary buffer. If TDES1[30] is set, it indicates the last buffer of the frame. †

After the last buffer of the frame has been transmitted, the DMA writes back the final status information to the Transmit Descriptor 0 (TDES0) word of the descriptor that has the last segment set in Transmit Descriptor 1 (TDES1[30]). At this time, if Interrupt on Completion (TDES1[31]) is set, the Bit 0 (Transmit Interrupt) of Register 5 (Status Register) is set, the Next descriptor is fetched, and the process repeats. †

The actual frame transmission begins after the MTL transmit FIFO buffer has reached either a programmable transmit threshold (Bits [16:14] of Register 6 (Operation Mode Register)), or a full frame is contained in the FIFO buffer. There is also an option for Store and Forward Mode (Bit 21 of Register 6 (Operation Mode Register)). Descriptors are released (Own bit TDES0[31] clears) when the DMA finishes transferring the frame. †

Note: To ensure proper transmission of a frame and the next frame, you must specify a non-zero buffer size for the transmit descriptor that has the Last Descriptor (TDES1[30]) set. †

Transmit Polling Suspended

Transmit polling can be suspended by either of the following conditions: †

- The DMA detects a descriptor owned by the Host (TDES0[31]=0). To resume, the driver must give descriptor ownership to the DMA and then issue a Poll Demand command. †
- A frame transmission is aborted when a transmit error because of underflow is detected. The appropriate Transmit Descriptor 0 (TDES0) bit is set. †

If the DMA goes into SUSPEND state because of the first condition, then both Bit 16 (Normal Interrupt Summary) and Bit 2 (Transmit Buffer Unavailable) of Register 5 (Status Register) are set. If the second condition occur, both Bit 15 (Abnormal Interrupt Summary) and Bit 5 (Transmit Underflow) of Register 5 (Status Register) are set, and the information is written to Transmit Descriptor 0, causing the suspension. †

In both cases, the position in the transmit List is retained. The retained position is that of the descriptor following the Last descriptor closed by the DMA. †

The driver must explicitly issue a Transmit Poll Demand command after rectifying the suspension cause. †

Reception

Receive functions use receive descriptors.

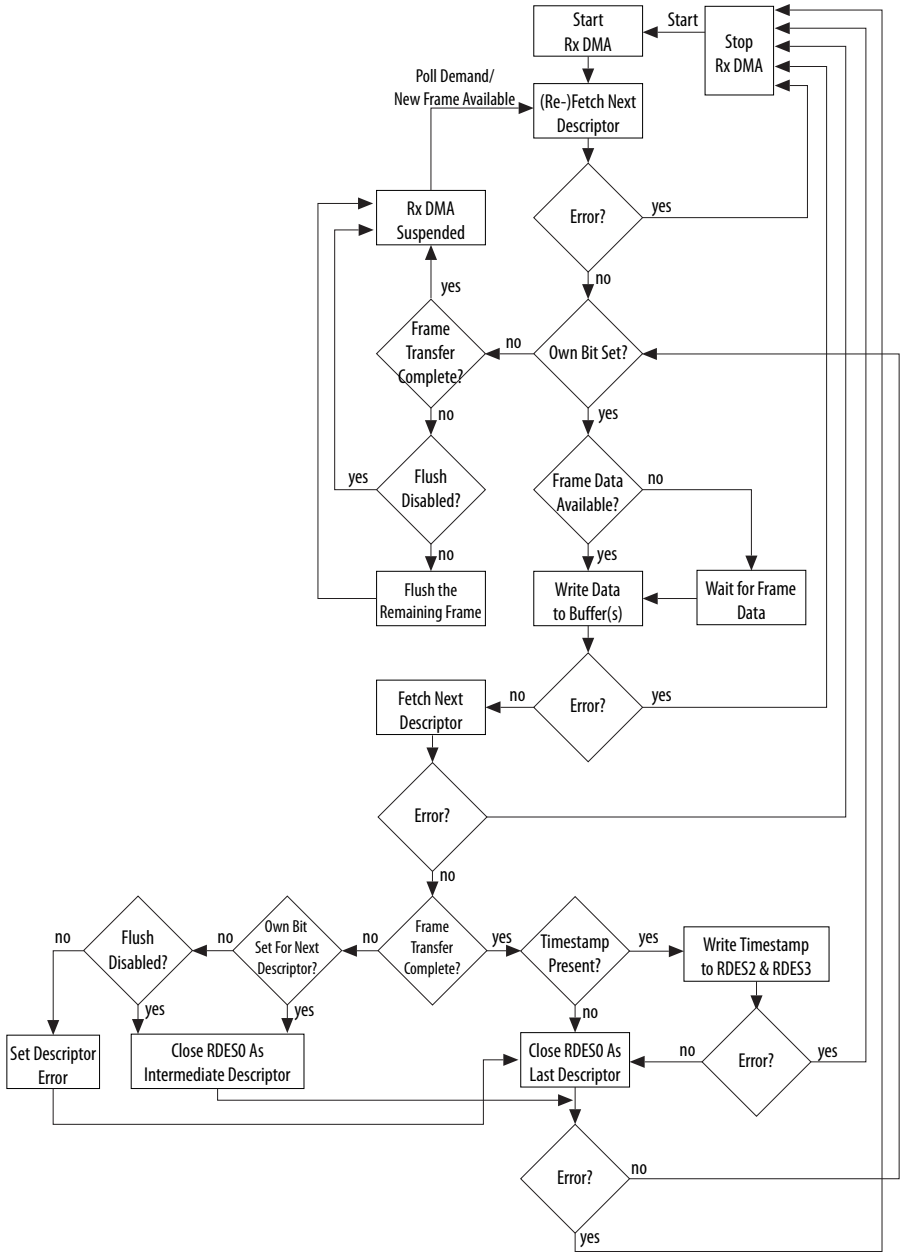
The receive DMA engine's reception sequence is proceeds as follows:

1. The host sets up receive descriptors (RDES0-RDES3) and sets the Own bit (RDES0[31]).†
2. When Bit 1 (SR) of Register 6 (Operation Mode Register) is set, the DMA enters the Run state. While in the Run state, the DMA polls the receive descriptor list, attempting to acquire free descriptors. If the fetched descriptor is not free (is owned by the host), the DMA enters the Suspend state and jumps to **step 9**.†
3. The DMA decodes the receive data buffer address from the acquired descriptors.†
4. Incoming frames are processed and placed in the acquired descriptor's data buffers.†
5. When the buffer is full or the frame transfer is complete, the receive engine fetches the next descriptor.†
6. If the current frame transfer is complete, the DMA proceeds to **step 7**. If the DMA does not own the next fetched descriptor and the frame transfer is not complete (EOF is not yet transferred), the DMA sets the Descriptor Error bit in the RDES0 (unless flushing is disabled in Bit 24 of Register 6 (Operation

Mode Register)). The DMA closes the current descriptor (clears the Own bit) and marks it as intermediate by clearing the Last Segment (LS) bit in the RDES0 value (marks it as Last Descriptor if flushing is not disabled), then proceeds to **step 8**. If the DMA does own the next descriptor but the current frame transfer is not complete, the DMA closes the current descriptor as intermediate and reverts to **step 4**.†

7. If IEEE 1588 timestamping is enabled, the DMA writes the timestamp (if available) to the current descriptor's RDES2 and RDES3. It then takes the receive frame's status from the MTL and writes the status word to the current descriptor's RDES0, with the Own bit cleared and the Last Segment bit set.†
8. The receive engine checks the latest descriptor's Own bit. If the host owns the descriptor (Own bit is 0), the Bit 7 (Receive Buffer Unavailable) of Register 5 (Status Register) is set and the DMA receive engine enters the Suspended state (Step 9). If the DMA owns the descriptor, the engine returns to **step 4** and awaits the next frame.
9. Before the receive engine enters the Suspend state, partial frames are flushed from the receive FIFO buffer. You can control flushing using Bit 24 of Register 6 (Operation Mode Register). †
10. The receive DMA exits the Suspend state when a Receive Poll demand is given or the start of next frame is available from the MTL's receive FIFO buffer. The engine proceeds to **step 2** and refetches the next descriptor. †

Figure 17-8: Receive DMA Operation



When software has enabled timestamping through the `tsena` bit of register 448 (Timestamp Control Register) and a valid timestamp value is not available for the frame (for example, because the receive FIFO buffer was full before the timestamp could be written to it), the DMA writes all ones to RDES2 and RDES3 descriptors . Otherwise (that is, if timestamping is not enabled), the RDES2 and RDES3 descriptors remain unchanged.

Receive Descriptor Acquisition

The receive Engine always attempts to acquire an extra descriptor in anticipation of an incoming frame. Descriptor acquisition is attempted if any of the following conditions is satisfied: †

- Bit 1 (Start or Stop Receive) of Register 6 (Operation Mode Register) has been set immediately after being placed in the Run state. †
- The data buffer of the current descriptor is full before the frame ends for the current transfer. †
- The controller has completed frame reception, but the current receive descriptor is not yet closed. †
- The receive process has been suspended because of a host-owned buffer (RDES0[31] = 0) and a new frame is received. †
- A Receive poll demand has been issued. †

Receive Frame Processing

The MAC transfers the received frames to the Host memory only when the frame passes the address filter and frame size is greater than or equal to the configurable threshold bytes set for the receive FIFO buffer of MTL, or when the complete frame is written to the FIFO buffer in store-and-forward mode. †

If the frame fails the address filtering, it is dropped in the MAC block itself (unless Bit 31 (Receive All) of Register 1 (MAC Frame Filter) is set). Frames that are shorter than 64 bytes, because of collision or premature termination, can be removed from the MTL receive FIFO buffer. †

After 64 (configurable threshold) bytes have been received, the MTL block requests the DMA block to begin transferring the frame data to the receive buffer pointed by the current descriptor. The DMA sets the First Descriptor (RDES0[9]) after the DMA Host interface becomes ready to receive a data transfer (if the DMA is not fetching transmit data from the host), to delimit the frame. The descriptors are released when the Own (RDES[31]) bit is clear, either as the Data buffer fills up or as the last segment of the frame is transferred to the receive buffer. If the frame is contained in a single descriptor, both Last Descriptor (RDES0[8]) and First Descriptor (RDES0[9]) are set.

The DMA fetches the next descriptor, sets the Last Descriptor (RDES[8]) bit, and releases the RDES0 status bits in the previous frame descriptor. Then the DMA sets bit 6 (Receive Interrupt) of Register 5 (Status Register). The same process repeats unless the DMA encounters a descriptor flagged as being owned by the host. If this occurs,Bit 7 (Receive Buffer Unavailable) of Register 5 (Status Register) is set and the receive process enters the Suspend state. The position in the receive list is retained. †

Receive Process Suspended

If a new receive frame arrives while the receive process is in the suspend state, the DMA refetches the current descriptor in the Host memory. If the descriptor is now owned by the DMA, the receive process re-enters the run state and starts frame reception. If the descriptor is still owned by the host, by default, the DMA discards the current frame at the top of the MTL RX FIFO buffer and increments the missed frame counter. If more than one frame is stored in the MTL EX FIFO buffer, the process repeats. †

The discarding or flushing of the frame at the top of the MTL EX FIFO buffer can be avoided by disabling Flushing (Bit 24 of Register 6 (Operation Mode Register)). In such conditions, the receive process sets the Receive Buffer Unavailable status and returns to the Suspend state. †

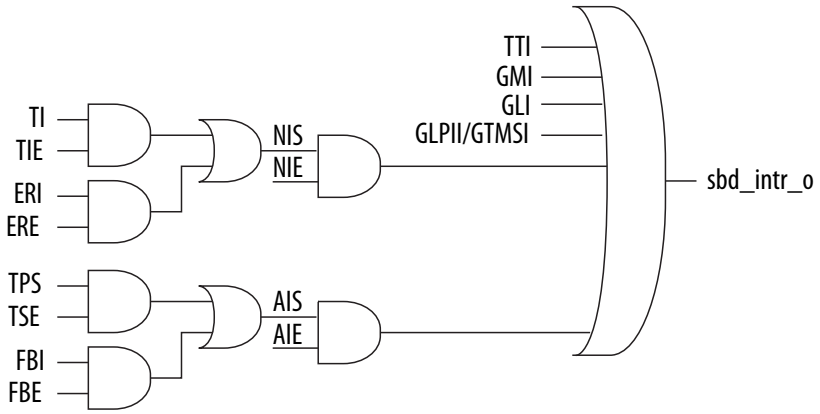
Interrupts

Interrupts can be generated as a result of various events. The DMA Register 5 (Status Register) contains a status bit for each of the events that can cause an interrupt. Register 7 (Interrupt Enable Register) contains an enable bit for each of the possible interrupt sources.

There are two groups of interrupts, Normal and Abnormal, as described in Register 5 (Status Register). Interrupts are cleared by writing a 1 to the corresponding bit position. When all the enabled interrupts

within a group are cleared, the corresponding summary bit is cleared. When both the summary bits are cleared, the sbd_intr_o interrupt signal is deasserted. If the MAC is the cause for assertion of the interrupt, then any of the GLI, GMI, TTI, or GLPII bits of Register 5 (Status Register) are set to 1.

Figure 17-9: Summary Interrupt (sbd_intr_o) Generation(51)



Note: Register 5 (Status Register) is the interrupt status register. The interrupt pin (sbd_intr_o) is asserted because of any event in this status register only if the corresponding interrupt enable bit is set in Register 7 (Interrupt Enable Register). †

Interrupts are not queued, and if the interrupt event occurs before the driver has responded to it, no additional interrupts are generated. For example, Bit 6 (Receive Interrupt) of Register 5 (Status Register) indicates that one or more frames were transferred to the Host buffer. The driver must scan all descriptors, from the last recorded position to the first one owned by the DMA.

An interrupt is generated only once for multiple, simultaneous events. The driver must scan Register 5 (Status Register) for the cause of the interrupt. After the driver has cleared the appropriate bit in Register 5 (Status Register), the interrupt is not generated again until a new interrupting event occurs. For example, the controller sets Bit 6 (Receive Interrupt) of Register 5 (Status Register) and the driver begins reading Register 5 (Status Register). Next, the interrupt indicated by Bit 7 (Receive Buffer Unavailable) of Register 5 (Status Register) occurs. The driver clears the receive interrupt (bit 6). However, the sbd_intr_o signal is not deasserted, because of the active or pending Receive Buffer Unavailable interrupt.

Bits 7:0 (riwt field) of Register 9 (Receive Interrupt Watchdog Timer Register) provide for flexible control of the receive interrupt. When this Interrupt timer is programmed with a non-zero value, it gets activated as soon as the RX DMA completes a transfer of a received frame to system memory without asserting the receive Interrupt because it is not enabled in the corresponding Receive Descriptor (RDES1[31]). When this timer runs out as per the programmed value, the AIS bit is set and the interrupt is asserted if the corresponding AIE is enabled in Register 7 (Interrupt Enable Register). This timer is disabled before it runs out, when a frame is transferred to memory, and the receive interrupt is triggered if it is enabled.

Related Information

Receive Descriptor on page 17-38

Error Response to DMA

If the slave replies with an error response to any data transfer initiated by a DMA channel, that DMA stops all operations and updates the error bits and the Fatal Bus Error bit in the Register 5 (Status Register). The

(51) Signals NIS and AIS are registered.

DMA controller can resume operation only after soft resetting or hard resetting the EMAC and reinitializing the DMA.

Descriptor Overview

The DMA in the Ethernet subsystem transfers data based on a single enhanced descriptor, as explained in the DMA Controller section. The enhanced descriptor is created in the system memory. The descriptor addresses must be word-aligned.

The enhanced or alternate descriptor format can have 8 DWORDS (32 bytes) instead of 4 DWORDS as in the case of the normal descriptor format.

The features of the enhanced or alternate descriptor structure are:

- The alternative descriptor structure is implemented to support buffers of up to 8 KB (useful for Jumbo frames).†
- There is a re-assignment of control and status bits in TDES0, TDES1, RDES0 (advanced timestamp or IPC full offload configuration), and RDES1. †
- The transmit descriptor stores the timestamp in TDES6 and TDES7 when you select the advanced timestamp. †
- The receive descriptor structure is also used for storing the extended status (RDES4) and timestamp (RDES6 and RDES7) when advanced timestamp, IPC Full Checksum Offload Engine, or Layer 3 and Layer 4 filter feature is selected. †
- You can select one of the following options for descriptor structure:
  - If timestamping is enabled in Register 448 (Timestamp Control Register) or Checksum Offload is enabled in Register 0 (MAC Configuration Register), the software must to allocate 32 bytes (8 DWORDS) of memory for every descriptor by setting Bit 7 (Descriptor Size) of Register 0 (Bus Mode Register).
  - If timestamping or Checksum Offload is not enabled, the extended descriptors (DES4 to DES7) are not required. Therefore, software can use descriptors with the default size of 16 bytes (4 DWORDS) by clearing Bit 7 (Descriptor Size) of Register 0 (Bus Mode Register) to 0.

Related Information

DMA Controller on page 17-18

Transmit Descriptor

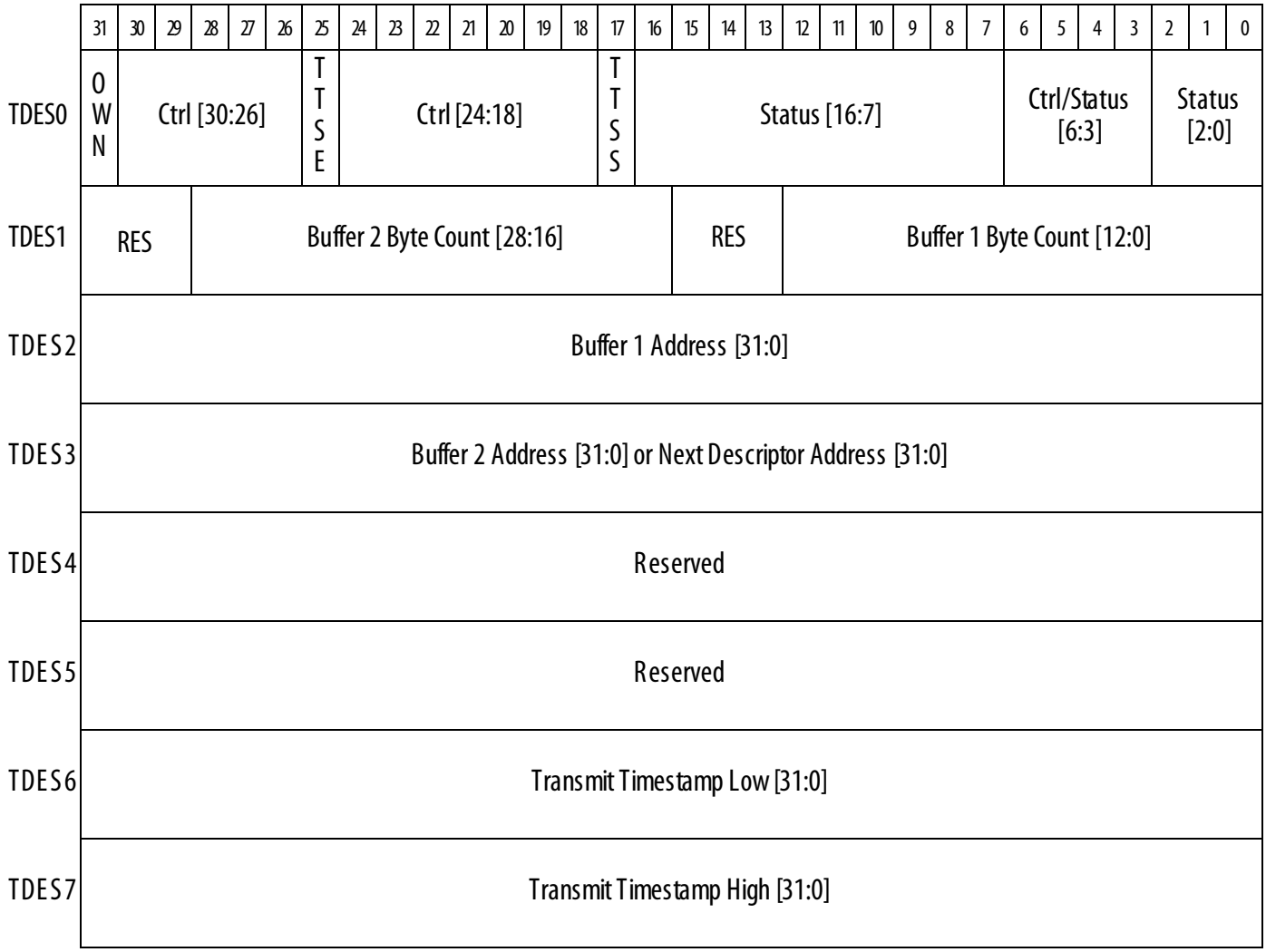
The application software must program the control bits TDES0[31:18] during the transmit descriptor initialization. When the DMA updates the descriptor, it writes back all the control bits except the OWN bit (which it clears) and updates the status bits[7:0].

With the advance timestamp support, the snapshot of the timestamp to be taken can be enabled for a given frame by setting Bit 25 (TTSE) of TDES0. When the descriptor is closed (that is, when the OWN bit is cleared), the timestamp is written into TDES6 and TDES7 as indicated by the status Bit 17 (TTSS) of TDES0.

Note: Only enhanced descriptor formats (4 or 8 DWORDS) are supported.

Note: When the advanced timestamp feature is enabled, software should set Bit 7 of Register 0 (Bus Mode Register), so that the DMA operates with extended descriptor size. When this control bit is clear, the TDES4-TDES7 descriptor space is not valid. †

Figure 17-10: Transmit Enhanced Descriptor Fields - Format



The DMA always reads or fetches four DWORDS of the descriptor from system memory to obtain the buffer and control information. †

Figure 17-11: Transmit Descriptor Fetch (Read) Format

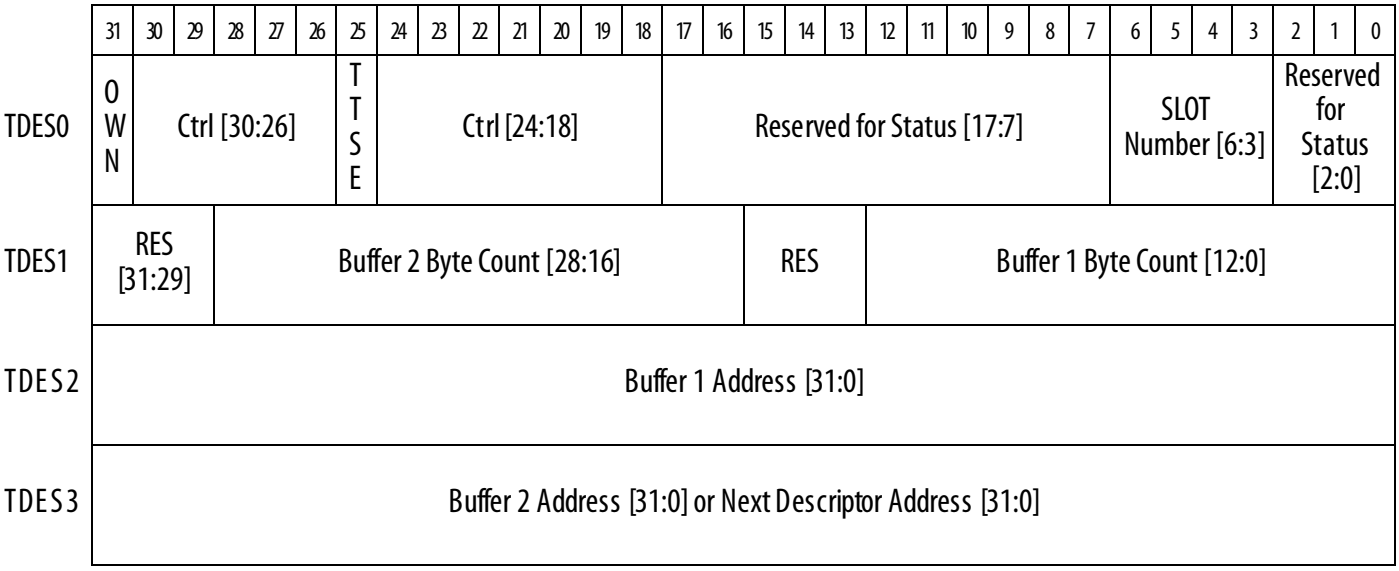


Table 17-6: Transmit Descriptor Word 0 (TDES0)

Bit	Description
31	OWN: Own Bit When set, this bit indicates that the descriptor is owned by the DMA. When this bit is cleared, it indicates that the descriptor is owned by the Host. The DMA clears this bit either when it completes the frame transmission or when the buffers allocated in the descriptor are read completely. The ownership bit of the frame's first descriptor must be set after all subsequent descriptors belonging to the same frame have been set to avoid a possible race condition between fetching a descriptor and the driver setting an ownership bit. †
30	IC: Interrupt on Completion When set, this bit enables the Transmit Interrupt (Register 5[0]) to be set after the present frame has been transmitted. †
29	LS: Last Segment When set, this bit indicates that the buffer contains the last segment of the frame. When this bit is set, the TBS1 or TBS2 field in TDES1 should have a non-zero value. †
28	FS: First Segment When set, this bit indicates that the buffer contains the first segment of a frame. †
27	DC: Disable CRC When this bit is set, the MAC does not append a CRC to the end of the transmitted frame. This bit is valid only when the first segment (TDES0[28]) is set. †

Bit	Description
26	DP: Disable Pad When set, the MAC does not automatically add padding to a frame shorter than 64 bytes. When this bit is cleared, the DMA automatically adds padding and CRC to a frame shorter than 64 bytes, and the CRC field is added despite the state of the DC (TDES0[27]) bit. This bit is valid only when the first segment (TDES0[28]) is set. †
25	TTSE: Transmit Timestamp Enable When set, this bit enables IEEE1588 hardware timestamping for the transmit frame referenced by the descriptor. This field is valid only when the First Segment control bit (TDES0[28]) is set.
24	Reserved
23:22	CIC: Checksum Insertion Control. These bits control the checksum calculation and insertion. The following list describes the bit encoding: ▪ 0x0: Checksum insertion disabled. ▪ 0x1: Only IP header checksum calculation and insertion are enabled. ▪ 0x2: IP header checksum and payload checksum calculation and insertion are enabled, but pseudoheader checksum is not calculated in hardware. ▪ 0x3: IP Header checksum and payload checksum calculation and insertion are enabled, and pseudoheader checksum is calculated in hardware. This field is valid when the First Segment control bit (TDES0[28]) is set.
21	TER: Transmit End of Ring When set, this bit indicates that the descriptor list reached its final descriptor. The DMA returns to the base address of the list, creating a descriptor ring. †
20	TCH: Second Address Chained When set, this bit indicates that the second address in the descriptor is the Next descriptor address rather than the second buffer address. When TDES0[20] is set, TBS2 (TDES1[28:16]) is a "don't care" value. TDES0[21] takes precedence over TDES0[20]. †
19:18	Reserved
17	TTSS: Transmit Timestamp Status This field is used as a status bit to indicate that a timestamp was captured for the described transmit frame. When this bit is set, TDES2 and TDES3 have a timestamp value captured for the transmit frame. This field is only valid when the descriptor's Last Segment control bit (TDES0[29]) is set. †

Bit	Description
16	IHE: IP Header Error When set, this bit indicates that the MAC transmitter detected an error in the IP datagram header. The transmitter checks the header length in the IPv4 packet against the number of header bytes received from the application and indicates an error status if there is a mismatch. For IPv6 frames, a header error is reported if the main header length is not 40 bytes. Furthermore, the Ethernet Length/Type field value for an IPv4 or IPv6 frame must match the IPheader version received with the packet. For IPv4 frames, an error status is also indicated if the Header Length field has a value less than 0x5. † This bit is valid only when the Tx Checksum Offload is enabled. If COE detects an IP header error, it still inserts an IPv4 header checksum if the Ethernet Type field indicates an IPv4 payload.†
15	ES: Error Summary Indicates the logical OR of the following bits: ▪ TDES0[14]: Jabber Timeout ▪ TDES0[13]: Frame Flush ▪ TDES0[11]: Loss of Carrier ▪ TDES0[10]: No Carrier ▪ TDES0[9]: Late Collision ▪ TDES0[8]: Excessive Collision ▪ TDES0[2]: Excessive Deferral ▪ TDES0[1]: Underflow Error ▪ TDES0[16]: IP Header Error ▪ TDES0[12]: IP Payload Error †
14	JT: Jabber Timeout When set, this bit indicates the MAC transmitter has experienced a jabber time-out. This bit is only set when Bit 22 (Jabber Disable) of Register 0 (MAC Configuration Register) is not set. †
13	FF: Frame Flushed When set, this bit indicates that the DMA or MTL flushed the frame because of a software Flush command given by the CPU. †
12	IPE: IP Payload Error When set, this bit indicates that MAC transmitter detected an error in the TCP, UDP, or ICMP IP datagram payload. The transmitter checks the payload length received in the IPv4 or IPv6 header against the actual number of TCP, UDP, or ICMP packet bytes received from the application and issues an error status in case of a mismatch. †

Bit	Description
11	LC: Loss of Carrier When set, this bit indicates that a loss of carrier occurred during frame transmission (that is, the gmii_crs_i signal was inactive for one or more transmit clock periods during frame transmission). This bit is valid only for the frames transmitted without collision when the MAC operates in the half-duplex mode. †
10	NC: No Carrier When set, this bit indicates that the Carrier Sense signal form the PHY was not asserted during transmission. †
9	LC: Late Collision When set, this bit indicates that frame transmission is aborted because of a collision occurring after the collision window (64 byte-times, including preamble, in MII mode and 512 byte-times, including preamble and carrier extension, in GMII mode). This bit is not valid if the Underflow Error bit is set.
8	EC: Excessive Collision When set, this bit indicates that the transmission was aborted after 16 successive collisions while attempting to transmit the current frame. If Bit 9 (Disable Retry) in Register 0 (MAC Configuration Register) is set, this bit is set after the first collision, and the transmission of the frame is aborted. †
7	VF: VLAN Frame When set, this bit indicates that the transmitted frame is a VLAN-type frame. †
6:3	CC: Collision Count (Status field) These status bits indicate the number of collisions that occurred before the frame was transmitted. This count is not valid when the Excessive Collisions bit (TDES0[8]) is set. The EMAC updates this status field only in the half-duplex mode.
2	ED: Excessive Deferral When set, this bit indicates that the transmission has ended because of excessive deferral of over 24,288 bit times (155,680 bits times in 1,000-Mbps mode or if Jumbo frame is enabled) if Bit 4 (Deferral Check) bit in Register 0 (MAC Configuration Register) is set. †
1	UF: Underflow Error When set, this bit indicates that the MAC aborted the frame because the data arrived late from the Host memory. Underflow Error indicates that the DMA encountered an empty transmit buffer while transmitting the frame. The transmission process enters the Suspended state and sets both Transmit Underflow (Register 5[5]) and Transmit Interrupt (Register 5[0]). †
0	DB: Deferred Bit When set, this bit indicates that the MAC defers before transmission because of the presence of carrier. This bit is valid only in the half-duplex mode. †

Table 17-7: Transmit Descriptor Word 1 (TDES1)

Bit	Description
31:29	Reserved
28:16	TBS2: Transmit Buffer 2 Size This field indicates the second data buffer size in bytes. This field is not valid if TDES0[20] is set. †
15:13	Reserved †
12:0	TBS1: Transmit Buffer 1 Size This field indicates the first data buffer byte size, in bytes. If this field is 0, the DMA ignores this buffer and uses Buffer 2 or the next descriptor, depending on the value of TCH (TDES0[20]). †

Table 17-8: Transmit Descriptor 2 (TDES2)

Bit	Description
31:0	Buffer 1 Address Pointer These bits indicate the physical address of Buffer 1. There is no limitation on the buffer address alignment. †

Table 17-9: Transmit Descriptor 3 (TDES3)

Bit	Description
31:0	Buffer 2 Address Pointer (Next Descriptor Address) Indicates the physical address of Buffer 2 when a descriptor ring structure is used. If the Second Address Chained (TDES0[20]) bit is set, this address contains the pointer to the physical memory where the Next descriptor is present. The buffer address pointer must be aligned to the bus width only when TDES0[20] is set. (LSBs are ignored internally.) †

Table 17-10: Transmit Descriptor 6 (TDES6)

Bit	Description
31:0	TTSL: Transmit Frame Timestamp Low This field is updated by DMA with the least significant 32 bits of the timestamp captured for the corresponding transmit frame. This field has the timestamp only if the Last Segment bit (LS) in the descriptor is set and Timestamp status (TTSS) bit is set. †

Table 17-11: Transmit Descriptor 7 (TDES7)

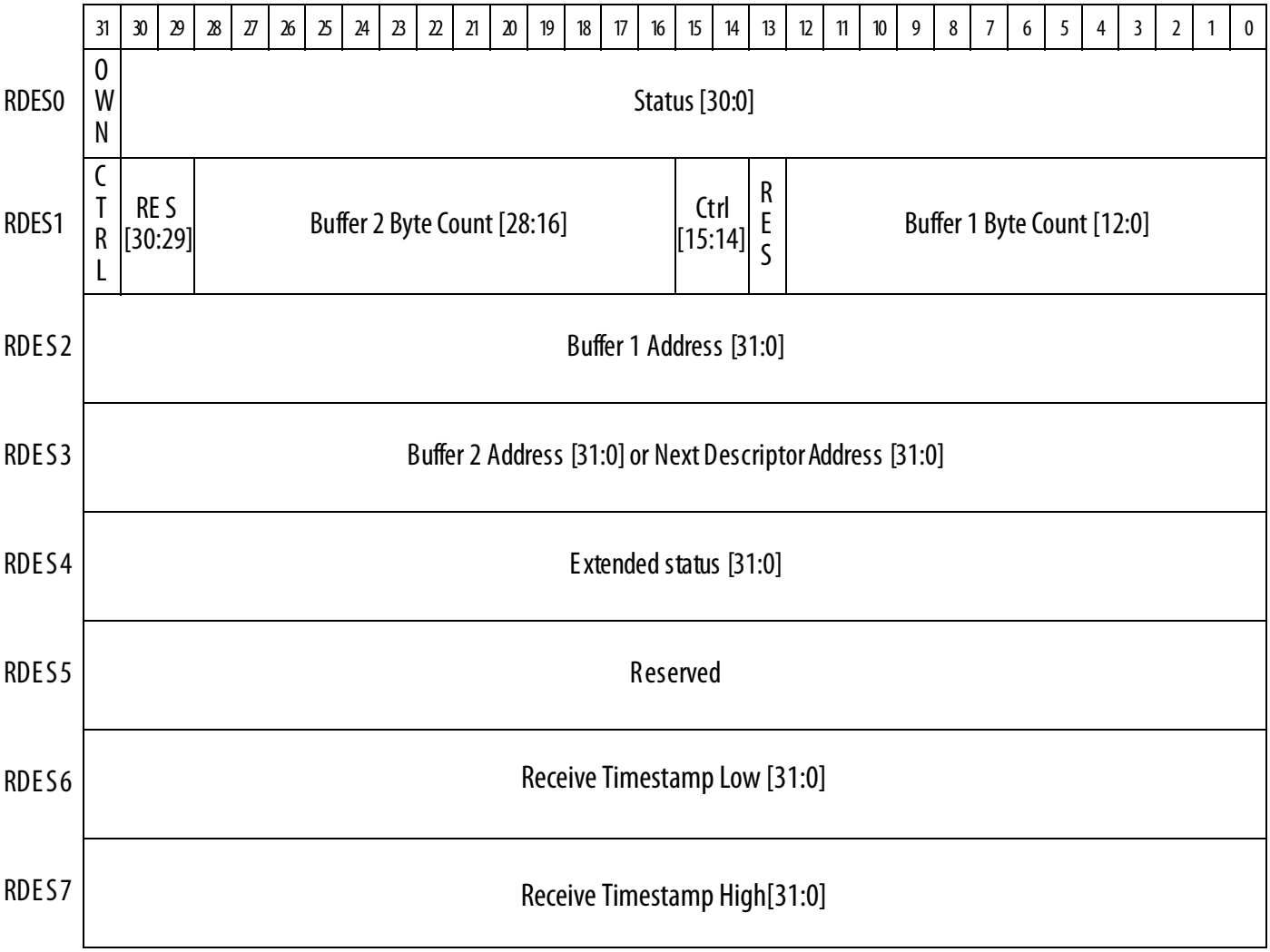
Bit	Description
31:0	TTSH: Transmit Frame Timestamp High This field is updated by DMA with the most significant 32 bits of the timestamp captured for the corresponding receive frame. This field has the timestamp only if the Last Segment bit (LS) in the descriptor is set and Timestamp status (TTSS) bit is set. †

Receive Descriptor

The receive descriptor can have 32 bytes of descriptor data (8 DWORDs) when advanced timestamp or IPC Full Offload feature is selected. When either of these features is enabled, software should set bit 7 of Register 0 (Bus Mode Register) so that the DMA operates with extended descriptor size. When this control bit is clear, the RDES0[0] is always cleared and the RDES4-RDES7 descriptor space is not valid. †

Note: Only enhanced descriptor formats (4 or 8 DWORDS) are supported.

Figure 17-12: Receive Enhanced Descriptor Fields Format



Receive Descriptor Field 0 (RDES0)

Table 17-12: Receive Descriptor Field 0 (RDES0)

Bit	Description
31	OWN: Own Bit When set, this bit indicates that the descriptor is owned by the DMA of the EMAC. When this bit is cleared, this bit indicates that the descriptor is owned by the Host. The DMA clears this bit either when it completes the frame reception or when the buffers that are associated with this descriptor are full.
30	AFM: Destination Address Filter Fail When set, this bit indicates a frame that failed in the DA Filter in the MAC. †
29:16	FL: Frame Length These bits indicate the byte length of the received frame that was transferred to host memory (including CRC). This field is valid when Last Descriptor (RDES0[8]) is set and either the Descriptor Error (RDES0[14]) or Overflow Error bits are cleared. The frame length also includes the two bytes appended to the Ethernet frame when IP checksum calculation (Type 1) is enabled and the received frame is not a MAC control frame. This field is valid when Last Descriptor (RDES0[8]) is set. When the Last Descriptor and Error Summary bits are not set, this field indicates the accumulated number of bytes that have been transferred for the current frame. †
15	ES: Error Summary Indicates the logical OR of the following bits: • RDES0[1]: CRC Error • RDES0[3]: Receive Error • RDES0[4]: Watchdog Timeout • RDES0[6]: Late Collision • RDES0[7]: Giant Frame • RDES4[4:3]: IP Header or Payload Error (Receive Descriptor Field 4 (RDES4)) • RDES0[11]: Overflow Error • RDES0[14]: Descriptor Error This field is valid only when the Last Descriptor (RDES0[8]) is set. †

Bit	Description
14	DE: Descriptor Error When set, this bit indicates a frame truncation caused by a frame that does not fit within the current descriptor buffers, and that the DMA does not own the Next descriptor. The frame is truncated. This bit is valid only when the Last Descriptor (RDES0[8]) bit is set. †
13	SAF: Source Address Filter Fail When set, this bit indicates that the SA field of frame failed the SA Filter in the MAC. †
12	LE: Length Error When set, this bit indicates that the actual length of the frame received and that the Length/ Type field does not match. This bit is valid only when the Frame Type (RDES0[5]) bit is clear. †
11	OE: Overflow Error When set, this bit indicates that the received frame was damaged because of buffer overflow in MTL. Note: This bit is set only when the DMA transfers a partial frame to the application, which happens only when the RX FIFO buffer is operating in the threshold mode. In the store-and-forward mode, all partial frames are dropped completely in the RX FIFO buffer. †
10	VLAN: VLAN Tag When set, this bit indicates that the frame to which this descriptor is pointing is a VLAN frame tagged by the MAC. The VLAN tagging depends on checking the VLAN fields of the received frame based on the Register 7 (VLAN Tag Register) setting. †
9	FS: First Descriptor When set, this bit indicates that this descriptor contains the first buffer of the frame. If the size of the first buffer is 0, the second buffer contains the beginning of the frame. If the size of the second buffer is also 0, the next descriptor contains the beginning of the frame. †
8	LD: Last Descriptor When set, this bit indicates that the buffers pointed to by this descriptor are the last buffers of the frame. †

Bit	Description
7	Timestamp Available When set, bit[7] indicates that a snapshot of the Timestamp is written in descriptor words 6 (RDES6) and 7 (RDES7). This is valid only when the Last Descriptor bit (RDES0[8]) is set.
6	LC: Late Collision When set, this bit indicates that a late collision has occurred while receiving the frame in the half-duplex mode. †
5	FT: Frame Type When set, this bit indicates that the receive frame is an Ethernet-type frame (the LT field is greater than or equal to 0x0600). When this bit is cleared, it indicates that the received frame is an IEEE802.3 frame. This bit is not valid for Runt frames less than 14 bytes.
4	RWT: Receive Watchdog Timeout When set, this bit indicates that the receive Watchdog Timer has expired while receiving the current frame and the current frame is truncated after the Watchdog Timeout. †
3	RE: Receive Error When set, this bit indicates that the `gmii_rxer_i` signal is asserted while `gmii_rxdv_i` is asserted during frame reception.
2	DE: Dribble Bit Error When set, this bit indicates that the received frame has a non-integer multiple of bytes (odd nibbles). This bit is valid only in the MII Mode. †
1	CE: CRC Error When set, this bit indicates that a CRC error occurred on the received frame. This bit is valid only when the Last Descriptor (RDES0[8]) is set. †

Bit	Description
0	Extended Status Available/RX MAC Address When either advanced timestamp or IP Checksum Offload (Type 2) is present, this bit, when set, indicates that the extended status is available in descriptor word 4 (RDES4). This bit is valid only when the Last Descriptor bit (RDES0[8]) is set. When the Advance Timestamp Feature or IPC Full Offload is not selected, this bit indicates RX MAC Address status. When set, this bit indicates that the RX MAC Address registers value (1 to 15) matched the frame's DA field. When clear, this bit indicates that the RX MAC Address Register 0 value matched the DA field. †

Related Information

- **Receive Descriptor Field 4 (RDES4)** on page 17-44
- **Receive Descriptor Field 6 (RDES6)** on page 17-47
- **Receive Descriptor Field 7 (RDES7)** on page 17-48

Receive Descriptor Field 1 (RDES1)

Table 17-13: Receive Descriptor Field 1 (RDES1)

Bit	Description
31	DIC: Disable Interrupt on Completion When set, this bit prevents setting the Status Register's RI bit (CSR5[6]) for the received frame ending in the buffer indicated by this descriptor. As a result, the RI interrupt for the frame is disabled and is not asserted to the Host.†
30:29	Reserved †
28:16	RBS2: Receive Buffer 2 Size These bits indicate the second data buffer size, in bytes. The buffer size must be a multiple of 4, even if the value of RDES3 (buffer2 address pointer) in the Receive Descriptor Field 3 (RDES3) is not aligned to the bus width. If the buffer size is not an appropriate multiple of 4, the resulting behavior is undefined. This field is not valid if RDES1[14] is set. For more information about calculating buffer sizes, refer to the Buffer Size Calculations section in this chapter.
15	RER: Receive End of Ring When set, this bit indicates that the descriptor list reached its final descriptor. The DMA returns to the base address of the list, creating a descriptor ring. †

Bit	Description
14	RCH: Second Address Chained When set, this bit indicates that the second address in the descriptor is the next descriptor address rather than the second buffer address. When this bit is set, RBS2 (RDES1[28:16]) is a "don't care" value. RDES1[15] takes precedence over RDES1[14]. †
13	Reserved †
12:0	RBS1: Receive Buffer 1 Size Indicates the first data buffer size in bytes. The buffer size must be a multiple of 4, even if the value of RDES2 (buffer1 address pointer), in the Receive Descriptor Field 2 (RDES2), is not aligned. When the buffer size is not a multiple of 4, the resulting behavior is undefined. If this field is 0, the DMA ignores this buffer and uses Buffer 2 or the next descriptor depending on the value of RCH (Bit 14). For more information about calculating buffer sizes, refer to the Buffer Size Calculations section in this chapter.

Related Information

- **Buffer Size Calculations** on page 17-21
- **Receive Descriptor Field 2 (RDES2)** on page 17-43
- **Receive Descriptor Field 3 (RDES3)** on page 17-44

Receive Descriptor Fields (RDES2) and (RDES3)

Receive Descriptor Field 2 (RDES2)

Table 17-14: Receive Descriptor Field 2 (RDES2)

Bit	Description
31:0	Buffer 1 Address Pointer These bits indicate the physical address of Buffer 1. There are no limitations on the buffer address alignment except for the following condition: The DMA uses the value programmed in RDES2[1:0] for its address generation when the RDES2 value is used to store the start of the frame. The DMA performs a write operation with the RDES2[1:0] bits as 0 during the transfer of the start of the frame but the frame is shifted as per the actual buffer address pointer. The DMA ignores RDES2[1:0] if the address pointer is to a buffer where the middle or last part of the frame is stored. For more information about buffer address alignment, refer to the *Host Data Buffer Alignment* section.

Related Information

Host Data Buffer Alignment on page 17-21

Receive Descriptor Field 3 (RDES3)

Table 17-15: Receive Descriptor Field 3 (RDES3)

Bit	Description
31:0	Buffer 2 Address Pointer (Next Descriptor Address) These bits indicate the physical address of Buffer 2 when a descriptor ring structure is used. If the Second Address Chained (RDES1[14]) bit in Receive Descriptor Field 1 (RDES1) is set, this address contains the pointer to the physical memory where the Next descriptor is present. If RDES1[14], in the Receive Descriptor Field 1 (RDES1) is set, the buffer (Next descriptor) address pointer must be bus width-aligned (RDES3[1:0] = 0. LSBs are ignored internally.) However, when RDES1[14] in the Receive Descriptor Field 1 (RDES1) is cleared, there are no limitations on the RDES3 value, except for the following condition: the DMA uses the value programmed in RDES3 [1:0] for its buffer address generation when the RDES3 value is used to store the start of frame. The DMA ignores RDES3 [1:0] if the address pointer is to a buffer where the middle or last part of the frame is stored.

Related Information

- **Host Data Buffer Alignment** on page 17-21
- **Receive Descriptor Field 1 (RDES1)** on page 17-42

Receive Descriptor Field 4 (RDES4)

The extended status is written only when there is status related to IPC or timestamp available. The availability of extended status is indicated by Bit 0 in RDES0. This status is available only when the Advance Timestamp or IPC Full Offload feature is selected.

Table 17-16: Receive Descriptor Field 4 (RDES4)

Bit	Description
31:28	Reserved †
27:26	Layer 3 and Layer 4 Filter Number Matched These bits indicate the number of the Layer 3 and Layer 4 Filter that matched the received frame. • 00: Filter 0 • 01: Filter 1 • 10: Filter 2 • 11: Filter 3 This field is valid only when Bit 24 or Bit 25 is set. When more than one filter matches, these bits give only the lowest filter number. †

Bit	Description
25	Layer 4 Filter Match When set, this bit indicates that the received frame matches one of the enabled Layer 4 Port Number fields. This status is given only when one of the following conditions is true: • Layer 3 fields are not enabled and all enabled Layer 4 fields match. • All enabled Layer 3 and Layer 4 filter fields match. When more than one filter matches, this bit gives the layer 4 filter status of filter indicated by Bits [27:26]. †
24	Layer 3 Filter Match When set, this bit indicates that the received frame matches one of the enabled Layer 3 IP Address fields. This status is given only when one of the following conditions is true: • All enabled Layer 3 fields match and all enabled Layer 4 fields are bypassed. • All enabled filter fields match. When more than one filter matches, this bit gives the layer 3 filter status of the filter indicated by Bits [27:26]. †
23:15	Reserved
14	Timestamp Dropped When set, this bit indicates that the timestamp was captured for this frame but got dropped in the MTL RX FIFO buffer because of overflow.
13	PTP Version When set, this bit indicates that the received PTP message has the IEEE 1588 version 2 format. When clear, it has the version 1 format.
12	PTP Frame Type When set, this bit indicates that the PTP message is sent directly over Ethernet. When this bit is not set and the message type is non-zero, it indicates that the PTP message is sent over UDP-IPv4 or UDP-IPv6. The information about IPv4 or IPv6 can be obtained from Bits 6 and 7.

Bit	Description
11:8	Message Type These bits are encoded to give the type of the message received. • 0000: No PTP message received • 0001: SYNC (all clock types) • 0010: Follow_Up (all clock types) • 0011: Delay_Req (all clock types) • 0100: Delay_Resp (all clock types) • 0101: Pdelay_Req (in peer-to-peer transparent clock) • 0110: Pdelay_Resp (in peer-to-peer transparent clock) • 0111: Pdelay_Resp_Follow_Up (in peer-to-peer transparent clock) • 1000: Announce • 1001: Management • 1010: Signaling • 1011-1110: Reserved • 1111: PTP packet with Reserved message type
7	IPv6 Packet Received When set, this bit indicates that the received packet is an IPv6 packet. This bit is updated only when Bit 10 (IPC) of Register 0 (MAC Configuration Register) is set.
6	IPv4 Packet Received When set, this bit indicates that the received packet is an IPv4 packet. This bit is updated only when Bit 10 (IPC) of Register 0 (MAC Configuration Register) is set.
5	IP Checksum Bypassed When set, this bit indicates that the checksum offload engine is bypassed.
4	IP Payload Error When set, this bit indicates that the 16-bit IP payload checksum (that is, the TCP, UDP, or ICMP checksum) that the EMAC calculated does not match the corresponding checksum field in the received segment. It is also set when the TCP, UDP, or ICMP segment length does not match the payload length value in the IP Header field. This bit is valid when either Bit 7 or Bit 6 is set.
3	IP Header Error When set, this bit indicates that either the 16-bit IPv4 header checksum calculated by the EMAC does not match the received checksum bytes, or the IP datagram version is not consistent with the Ethernet Type value. This bit is valid when either Bit 7 or Bit 6 is set.

Bit	Description
2:0	IP Payload Type These bits indicate the type of payload encapsulated in the IP datagram processed by the receive Checksum Offload Engine (COE). The COE also sets these bits to 0 if it does not process the IP datagram's payload due to an IP header error or fragmented IP. • 0x0: Unknown or did not process IP payload • 0x1: UDP • 0x2: TCP • 0x3: ICMP • 0x4–0x7: Reserved This bit is valid when either Bit 7 or Bit 6 is set.

Related Information

Receive Descriptor Field 0 (RDES0) on page 17-39

Receive Descriptor Fields (RDES6) and (RDES7)

Receive Descriptor Fields 6 and 7 (RDES6 and RDES7) contain the snapshot of the timestamp. The availability of the snapshot of the timestamp in RDES6 and RDES7 is indicated by Bit 7 in the RDES0 descriptor.

Related Information

Receive Descriptor Field 0 (RDES0) on page 17-39

Receive Descriptor Field 6 (RDES6)

Table 17-17: Receive Descriptor Field 6 (RDES6)

Bit	Description
31:0	RTSL: Receive Frame Timestamp Low This field is updated by the DMA with the least significant 32 bits of the timestamp captured for the corresponding receive frame. This field is updated by the DMA only for the last descriptor of the receive frame, which is indicated by Last Descriptor status bit (RDES0[8]) in RDES0.

Related Information

Receive Descriptor Field 0 (RDES0) on page 17-39

Receive Descriptor Field 7 (RDES7)

Table 17-18: Receive Descriptor Field 7 (RDES7)

Bit	Description
31:0	RTSH: Receive Frame Timestamp High This field is updated by the DMA with the most significant 32 bits of the timestamp captured for the corresponding receive frame. This field is updated by the DMA only for the last descriptor of the receive frame, which is indicated by Last Descriptor status bit (RDES0[8]) in RDES0.

Related Information

Receive Descriptor Field 0 (RDES0) on page 17-39

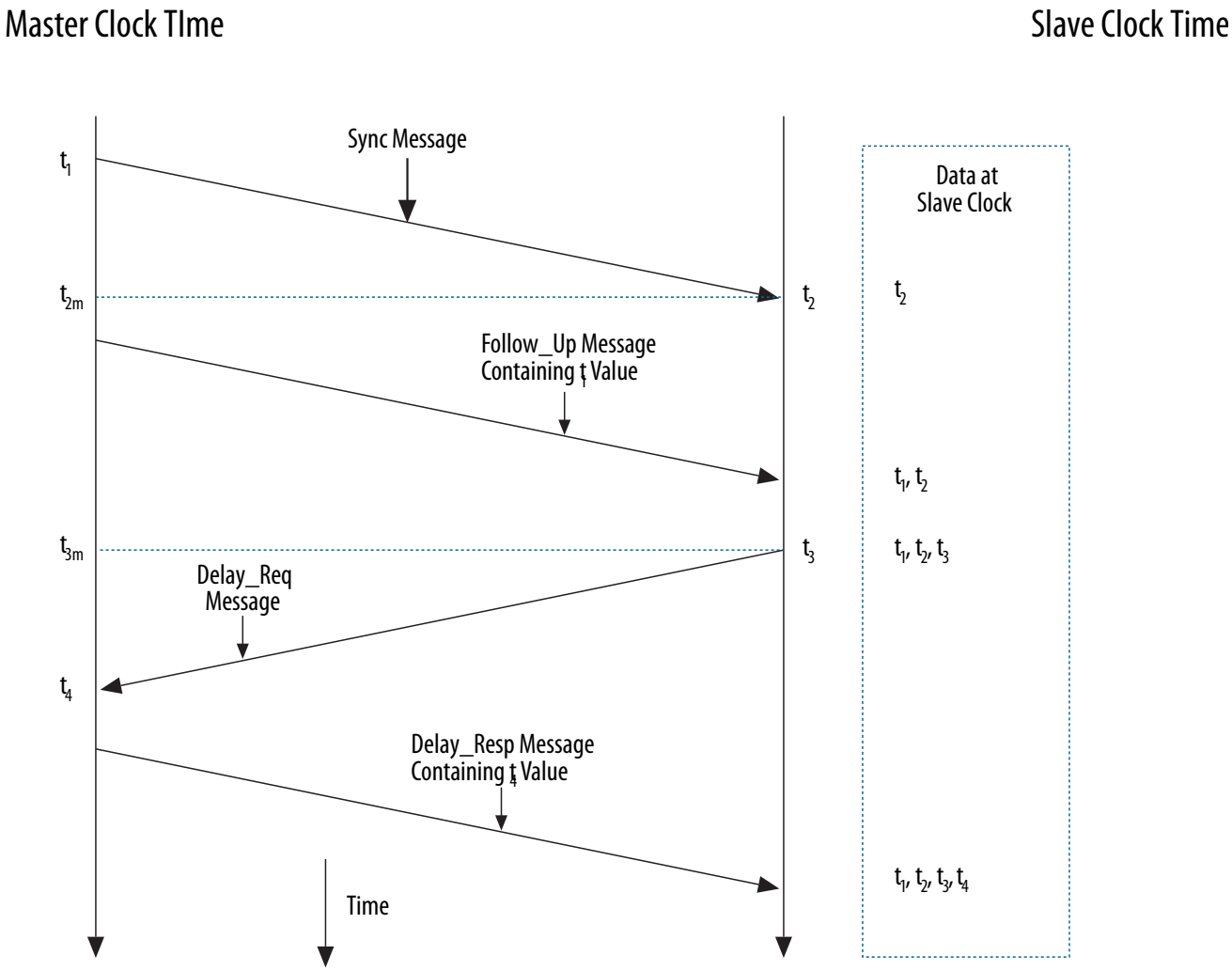
IEEE 1588-2002 Timestamps

The IEEE 1588-2002 standard defines the Precision Time Protocol (PTP) that enables precise synchronization of clocks in a distributed network of devices. The PTP applies to systems communicating by local area networks supporting multicast messaging. This protocol enables heterogeneous systems that include clocks of varying inherent precision, resolution, and stability to synchronize. It is frequently used in automation systems where a collection of communicating machines such as robots must be synchronized and hence operate over a common time base.(†)

The PTP is transported over UDP/IP. The system or network is classified into Master and Slave nodes for distributing the timing and clock information.†

The following figure shows the process that PTP uses for synchronizing a slave node to a master node by exchanging PTP messages.

Figure 17-13: Networked Time Synchronization



The PTP uses the following process for synchronizing a slave node to a master node by exchanging the PTP messages:

1. The master broadcasts the PTP Sync messages to all its nodes. The Sync message contains the master's reference time information. The time at which this message leaves the master's system is t1. This time must be captured, for Ethernet ports, at the PHY interface.†
2. The slave receives the sync message and also captures the exact time, t2, using its timing reference.†
3. The master sends a follow_up message to the slave, which contains t1 information for later use.†
4. The slave sends a delay_req message to the master, noting the exact time, t3, at which this frame leaves the PHY interface.†
5. The master receives the message, capturing the exact time, t4, at which it enters its system.†
6. The master sends the t4 information to the slave in the delay_resp message.†
7. The slave uses the four values of t1, t2, t3, and t4 to synchronize its local timing reference to the master's timing reference.†

Most of the PTP implementation is done in the software above the UDP layer. However, the hardware support is required to capture the exact time when specific PTP packets enter or leave the Ethernet port at

the PHY interface. This timing information must be captured and returned to the software for the proper implementation of PTP with high accuracy.†

The EMAC is intended to support IEEE 1588 operation in all modes with a resolution of 10 ns. When the three EMACs are operating in an IEEE 1588 environment, the MPU subsystem is responsible for maintaining synchronization between the time counters internal to the three MACs.

The IEEE 1588 interface to the FPGA allows the FPGA to provide a source for the `emac_ptp_ref_clk` input as well to allow it to monitor the pulse per second output from each EMAC controller.

The EMAC component provides a hardware assisted implementation of the IEEE 1588 protocol. Hardware support is for timestamp maintenance. Timestamps are updated when receiving any frame on the PHY interface, and the receive descriptor is updated with this value. Timestamps are also updated when the SFD of a frame is transmitted and the transmit descriptor is updated accordingly.†

Related Information

IEEE Standards Association

For details about the IEEE 1588-2002 standard, refer to IEEE Standard 1588-2002 - IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems, available on the IEEE Standards Association website.†

Reference Timing Source

To get a snapshot of the time, the EMAC takes the reference clock input and uses it to generate the reference time (64-bit) internally and capture timestamps.†

System Time Register Module

The 64-bit time is maintained in this module and updated using the input reference clock, `clk_ptp_ref`, which can be the `emac_ptp_clk` from the HPS or the `f2s_ptp_ref_clk` from the FPGA. The `emac_ptp_clk` in the HPS is a derivative of the `osc1_clk` and is configured in the clock manager. This input reference clock is the source for taking snapshots (timestamps) of Ethernet frames being transmitted or received at the PHY interface.

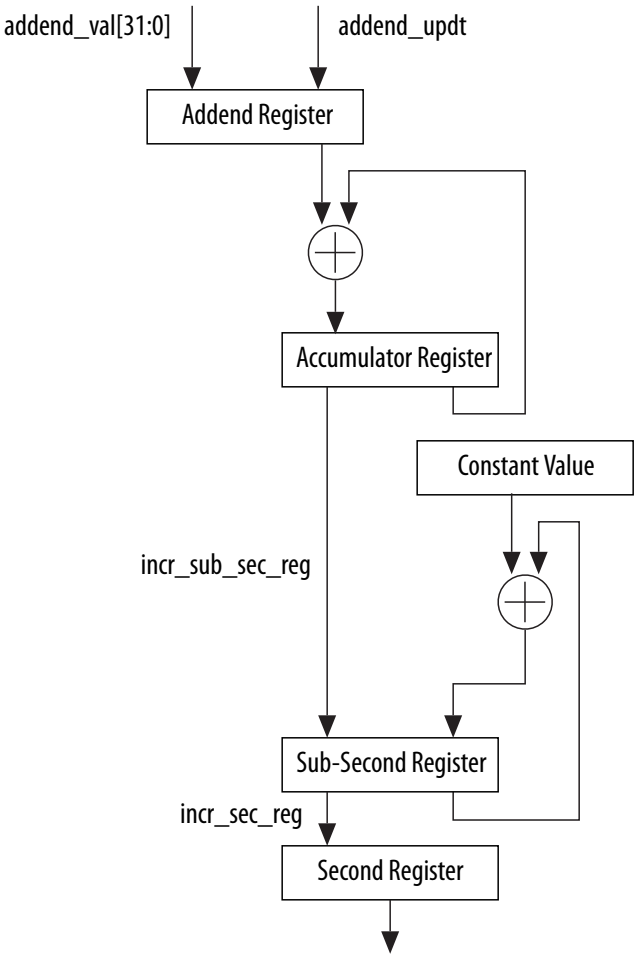
The system time counter can be initialized or corrected using the coarse correction method. In this method, the initial value or the offset value is written to the Timestamp Update register. For initialization, each EMAC's system time counter is written with the value in the Timestamp Update registers, while for system time correction, the offset value is added to or subtracted from the system time.

With the fine correction method, a slave clock's frequency drift with respect to the master clock is corrected over a period of time instead of in one clock, as in coarse correction. This protocol helps maintain linear time and does not introduce drastic changes (or a large jitter) in the reference time between PTP sync message intervals.†

With this method, an accumulator sums up the contents of the Timestamp_Addend register, as shown in the figure below. The arithmetic carry that the accumulator generates is used as a pulse to increment the system time counter. The accumulator and the addend are 32-bit registers. Here, the accumulator acts as a high-precision frequency multiplier or divider.

Note: You must connect a PTP clock with a frequency higher than the frequency required for the specified accuracy.†

Figure 17-14: Algorithm for System Time Update Using Fine Method



The System Time Update logic requires a 50-MHz clock frequency to achieve 20-ns accuracy. The frequency division ratio (FreqDivisionRatio) is the ratio of the reference clock frequency to the required clock frequency. Hence, if the reference clock (`clk_ptp_ref_i`) is for example, 66 MHz, this ratio is calculated as 66 MHz / 50 MHz = 1.32. Hence, the default addend value to program in the register is 232 / 1.32, 0xC1F07C1F.

If the reference clock drifts lower, to 65 MHz for example, the ratio is 65 / 50, or 1.3 and the value to set in the addend register is 232 / 1.30, or 0xC4EC4EC4. If the clock drifts higher, to 67 MHz for example, the addend register must be set to 0xBF0B7672. When the clock drift is nil, the default addend value of 0xC1F07C1F (232 / 1.32) must be programmed.†

In the above figure, the constant value used to accumulate the sub-second register is decimal 43, which achieves an accuracy of 20 ns in the system time (in other words, it is incremented in 20-ns steps).

The software must calculate the drift in frequency based on the Sync messages and update the Addend register accordingly.†

Initially, the slave clock is set with `FreqCompensationValue0` in the Addend register. This value is as follows:†

$$FreqCompensationValue_0 = 232 / FreqDivisionRatio^{\dagger}$$

If MasterToSlaveDelay is initially assumed to be the same for consecutive sync messages, the algorithm described below must be applied. After a few sync cycles, frequency lock occurs. The slave clock can then determine a precise MasterToSlaveDelay value and re-synchronize with the master using the new value.†

The algorithm is as follows:†

- At time $MasterSyncTime_n$ the master sends the slave clock a sync message. The slave receives this message when its local clock is $SlaveClockTime_n$ and computes $MasterClockTime_n$ as:†

  $$MasterClockTime_n = MasterSyncTime_n + MasterToSlaveDelay_n^{\dagger}$$

- The master clock count for current sync cycle, $MasterClockCount_n$ is given by:†

  $$MasterClockCount_n = MasterClockTime_n - MasterClockTime_{n-1}$$

  (assuming that MasterToSlaveDelay is the same for sync cycles n and n – 1)†

- The slave clock count for current sync cycle, $SlaveClockCount_n$ is given by:†

  $$SlaveClockCount_n = SlaveClockTime_n - SlaveClockTime_{n-1}^{\dagger}$$

- The difference between master and slave clock counts for current sync cycle, $ClockDiffCount_n$ is given by:†

  $$ClockDiffCount_n = MasterClockCount_n - SlaveClockCount_n^{\dagger}$$

- The frequency-scaling factor for the slave clock, $FreqScaleFactor_n$ is given by:†

  $$FreqScaleFactor_n = (MasterClockCount_n + ClockDiffCount_n) / SlaveClockCount_n^{\dagger}$$

- The frequency compensation value for Addend register, $FreqCompensationValue_n$ is given by: †

  $$FreqCompensationValue_n = FreqScaleFactor_n \times FreqCompensationValue_{n-1} - 1^{\dagger}$$

In theory, this algorithm achieves lock in one Sync cycle; however, it may take several cycles, because of changing network propagation delays and operating conditions.†

This algorithm is self-correcting: if for any reason the slave clock is initially set to a value from the master that is incorrect, the algorithm corrects it at the cost of more Sync cycles.†

Transmit Path Functions

The MAC captures a timestamp when the start-of-frame data (SFD) is sent on the PHY interface. You can control the frames for which timestamps are captured on a per frame basis. In other words, each transmit frame can be marked to indicate whether a timestamp should be captured for that frame. †

You can use the conrol bits in the transmit descriptor to indicate whether a timestamp should be capture for a frame. The MAC returns the timestamp to the software inside the corrsponding transmit descriptor,

thus connecting the timestamp automatically to the specific PTP frame. The 64-bit timestamp information is written to the TDES2 and TDES3 fields. T†

Receive Path Functions

The MAC captures the timestamp of all frames received on the PHY interface. The DMA returns the timestamp to the software in the corresponding receive descriptor. The timestamp is written only to the last receive descriptor.†

Timestamp Error Margin

According to the IEEE1588 specifications, a timestamp must be captured at the SFD of the transmitted and received frames at the PHY interface. Because the PHY interface receive and transmit clocks are not synchronous to the reference timestamp clock (`clk_ptp_ref`) a small amount of drift is introduced when a timestamp is moved between asynchronous clock domains. In the transmit path, the captured and reported timestamp has a maximum error margin of two PTP clocks, meaning that the captured timestamp has a reference timing source value that is occurred within two clocks after the SFD is transmitted on the PHY interface.

Similarly, in the receive path, the error margin is three PHY interface clocks, plus up to two PTP clocks. You can ignore the error margin due to the PHY interface clock by assuming that this constant delay is present in the system (or link) before the SFD data reaches the PHY interface of the MAC.†

Frequency Range of Reference Timing Clock

The timestamp information is transferred across asynchronous clock domains, from the EMAC clock domain to the FPGA clock domain. Therefore, a minimum delay is required between two consecutive timestamp captures. This delay is four PHY interface clock cycles and three PTP clock cycles. If the delay between two timestamp captures is less than this amount, the MAC does not take a timestamp snapshot for the second frame.

The maximum PTP clock frequency is limited by the maximum resolution of the reference time (20 ns resulting in 50 MHz) and the timing constraints achievable for logic operating on the PTP clock. In addition, the resolution, or granularity, of the reference time source determines the accuracy of the synchronization. Therefore, a higher PTP clock frequency gives better system performance.†

The minimum PTP clock frequency depends on the time required between two consecutive SFD bytes. Because the PHY interface clock frequency is fixed by the IEEE 1588 specification, the minimum PTP clock frequency required for proper operation depends on the operating mode and operating speed of the MAC.†

Table 17-19: Minimum PTP Clock Frequency Example

Mode	Minimum Gap Between Two SFDs	Minimum PTP Frequency
100-Mbps full-duplex operation	168 MII clocks (128 clocks for a 64-byte frame + 24 clocks of min IFG + 16 clocks of preamble)	(3 * PTP) + (4 * MII) <= 168 * MII, that is, ~0.5 MHz (168 – 4) * 40 ns ÷ 3 = 2180 ns period)

Mode	Minimum Gap Between Two SFDs	Minimum PTP Frequency
1000-Mbps half duplex operation	24 GMII clocks (4 for a jam pattern sent just after SFD because of collision + 12 IFG + 8 preamble)	(3 * PTP) + 4 * GMII <= 24 * GMII, that is, 18.75 MHz

Related Information

IEEE Standards Association

For details about jam patterns, refer to the *IEEE Std 802.3 2008 Part 3: Carrier sense multiple access with Collision Detection (CSMA/CD) Access Method and Physical Layer Specifications*, available on the IEEE Standards Association website.

IEEE 1588-2008 Advanced Timestamps

In addition to the basic timestamp features mentioned in IEEE 1588-2002 Timestamps, the EMAC supports the following advanced timestamp features defined in the IEEE 1588-2008 standard.†

- Supports the IEEE 1588-2008 (version 2) timestamp format. †
- Provides an option to take a timestamp of all frames or only PTP-type frames. †
- Provides an option to take a timestamp of event messages only. †
- Provides an option to take the timestamp based on the clock type: ordinary, boundary, end-to-end, or peer-to-peer. †
- Provides an option to configure the EMAC to be a master or slave for ordinary and boundary clock. †
- Identifies the PTP message type, version, and PTP payload in frames sent directly over Ethernet and sends the status. †
- Provides an option to measure sub-second time in digital or binary format. †

Related Information

IEEE Standards Association

For more information about advanced timestamp features, refer to the *IEEE Standard 1588 - 2008 IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control System, available on the IEEE Standards Association website.*

Peer-to-Peer PTP Transparent Clock (P2P TC) Message Support

The IEEE 1588-2008 version supports Peer-to-Peer PTP (Pdelay) messages in addition to SYNC, Delay Request, Follow-up, and Delay Response messages.†

Clock Types

The EMAC supports the following clock types defined in the IEEE 1588-2008 standard:

- Ordinary clock†
- Boundary clock†
- End-to-End transparent clock†
- Peer-to-Peer transparent clock†

Ordinary Clock

The ordinary clock in a domain supports a single copy of the protocol. The ordinary clock has a single PTP state and a single physical port. In typical industrial automation applications, an ordinary clock is associated with an application device such as a sensor or an actuator. In telecom applications, the ordinary clock can be associated with a timing demarcation device. †

The ordinary clock can be a grandmaster or a slave clock. It supports the following features:†

- Sends and receives PTP messages. The timestamp snapshot can be controlled as described in the `Timestamp Control (gmacgrp_timestamp_control)` register.†
- Maintains the data sets such as timestamp values.†

The table below shows the messages for which you can take the timestamp snapshot on the receive side for Master and slave nodes. For an ordinary clock, you can take the snapshot of either of the following PTP message types: version 1 or version 2. You cannot take the snapshots for both PTP message types. You can take the snapshot by setting the control bit (`tsver2ena`) and selecting the snapshot mode in the `Timestamp Control (gmacgrp_timestamp_control)` register.†

Table 17-20: Ordinary Clock: PTP Messages for Snapshot†

Master	Slave
Delay_Req	SYNC

Boundary Clock

The boundary clock typically has several physical ports communicating with the network. The messages related to synchronization, master-slave hierarchy, and signaling terminate in the protocol engine of the boundary clock and are not forwarded. The PTP message type status given by the MAC helps you to identify the type of message and take appropriate action. The boundary clock is similar to the ordinary clock except for the following features:†

- The clock data sets are common to all ports of the boundary clock .†
- The local clock is common to all ports of the boundary clock. Therefore, the features of the ordinary clock are also applicable to the boundary clock.†

End-to-End Transparent Clock

The end-to-end transparent clock supports the end-to-end delay measurement mechanism between slave clocks and the master clock. The end-to-end transparent clock forwards all messages like normal bridge, router, or repeater. The residence time of a PTP packet is the time taken by the PTP packet from the ingess port to the egress port.†

The residence time of a SYNC packet inside the end-to-end transparent clock is updated in the correction field of the associated Follow_Up PTP packet before it is transmitted. Similarly, the residence time of a Delay_Req packet inside the end-to-end transparent clock is updated in the correction field of the associated Delay_Resp PTP packet before it is transmitted. Therefore, the snapshot needs to be taken at both ingress and egress ports only for PTP messages SYNC or Delay_req. You can take the snapshot by setting the snapshot select bits (`SNAPTYPSEL`) to b'10 in the `Timestamp Control (gmacgrp_timestamp_control)` register.†

The `snaptypsel` bits, along with bits 15 and 14 in the `Timestamp Control` register, decide the set of PTP packet types for which a snapshot needs to be taken. The encoding is shown in the table below:†

Table 17-21: Timestamp Snapshot Dependency on Register Bits†

X is defined as a "don't care" in the table.

snaptypsel (bits[17:16])	tsmstrena (bit 15)	tsevntena (bit 14)	PTP Messages
0x0	X	0	SYNC, Follow_Up, Delay_Req, Delay_Resp
0x0	0	1	SYNC
0x0	1	1	Delay_Req
0x1	X	0	SYNC, Follow_Up, Delay_Req, Delay_Resp, Pdelay_Req, Pdelay_Resp, Pdelay_Resp_ Follow_Up
0x1	0	1	SYNC, Pdelay_Req, Pdelay_Resp
0x1	1	1	Delay_Req, Pdelay_Req, Pdelay_ Resp
0x2	X	X	SYNC, Delay_Req
0x3	X	X	Pdelay_Req, Pdelay_Resp

Peer-to-Peer Transparent Clock

The peer-to-peer transparent clock differs from the end-to-end transparent clock in the way it corrects and handles the PTP timing messages. In all other aspects, it is identical to the end-to-end transparent clock.†

In the peer-to-peer transparent clock, the computation of the link delay is based on an exchange of Pdelay_Req, Pdelay_Resp, and Pdelay_Resp_Follow_Up messages with the link peer. The residence time of the Pdelay_Req and the associated Pdelay_resp packets is added and inserted into the correction field of the associated Pdely_Resp_Followup packet.†

Therefore, support for taking snapshot for the event messages related to Pdelay is added as shown in the table below.†

Table 17-22: Peer-to-Peer Transparent Clock: PTP Messages for Snapshot†

PTP Messages
SYNC
Pdelay_Req
Pdelay_Resp

You can take the snapshot by setting the snapshot select bits (`snaptypsel`) to b'11 in the `Timestamp Control` register.†

Reference Timing Source

The EMAC supports the following reference timing source features defined in the IEEE 1588-2008 standard:

- 48-bit seconds field†
- Fixed pulse-per-second output†
- Flexible pulse-per-second output†
- Auxiliary snapshots (timestamps) with external events

Transmit Path Functions

The advanced timestamp feature is supported through the descriptors format.

Receive Path Functions

The MAC processes the received frames to identify valid PTP frames. You can control the snapshot of the time to be sent to the application, by using the following options:†

- Enable timestamp for all frames.†
- Enable timestamp for IEEE 1588 version 2 or version 1 timestamp.†
- Enable timestamp for PTP frames transmitted directly over Ethernet or UDP/IP Ethernet.†
- Enable timestamp snapshot for the received frame for IPv4 or IPv6.†
- Enable timestamp snapshot for EVENT messages (SYNC, DELAY_REQ, PDELAY_REQ, or PDELAY_RESP) only.†
- Enable the node to be a master or slave and select the timestamp type to control the type of messages for which timestamps are taken.†

The DMA returns the timestamp to the software inside the corresponding transmit or receive descriptor.

Auxiliary Snapshot

The auxiliary snapshot feature allows you to store a snapshot (timestamp) of the system time based on an external event. The event is considered to be the rising edge of the sideband signal `ptp_aux_ts_trig_i` from the FPGA. One auxiliary snapshot input is available. The depth of the auxiliary snapshot FIFO buffer is 16.

The timestamps taken for any input are stored in a common FIFO buffer. The host can read Register 458 (Timestamp Status Register) to know which input's timestamp is available for reading at the top of this FIFO buffer.

Only 64-bits of the timestamp are stored in the FIFO. You can read the upper 16-bits of seconds from Register 457 (System Time - Higher Word Seconds Register) when it is present. When a snapshot is stored, the MAC indicates this to the host with an interrupt. The value of the snapshot is read through a FIFO register access. If the FIFO becomes full and an external trigger to take the snapshot is asserted, then a snapshot trigger-missed status (`ATSSTM`) is set in Register 458 (Timestamp Status Register). This indicates that the latest auxiliary snapshot of the timestamp was not stored in the FIFO. The latest snapshot is not written to the FIFO when it is full. When a host reads the 64-bit timestamp from the FIFO, the space becomes available to store the next snapshot. You can clear a FIFO by setting Bit 19 (`ATSFC`) in Register 448 (Timestamp Control Register). When multiple snapshots are present in the FIFO, the count is indicated in Bits [27:25], `ATSNS`, of Register 458 (Timestamp Status Register). †

IEEE 802.3az Energy Efficient Ethernet

Energy Efficient Ethernet (EEE) standardized by IEEE 802.3-az, version D2.0 is supported by the EMAC. It is supported by the MAC operating in 10/100/1000 Mbps rates. EEE is only supported when the EMAC

is configured to operate with the RGMII PHY interface operating in full-duplex mode. It cannot be used in half-duplex mode.

EEE enables the MAC to operate in Low-Power Idle (LPI) mode. Either end point of an Ethernet link can disable functionality to save power during periods of low link utilization. The MAC controls whether the system should enter or exit LPI mode and communicates this information to the PHY.†

Related Information

IEEE 802.3 Ethernet Working Group

For details about the *IEEE 802.3az Energy Efficient Ethernet standard*, refer to the IEEE 802.3 Ethernet Working Group website.†

LPI Timers

Two timers internal to the EMAC are associated with LPI mode:

- LPI Link Status (LS) Timer†
- LPI Time Wait (TW) Timer†

The LPI LS timer counts, in ms, the time expired since the link status has come up. This timer is cleared every time the link goes down and is incremented when the link is up again and the terminal count as programmed by the software is reached. The PHY interface does not assert the LPI pattern unless the terminal count is reached. This protocol ensures a minimum time for which no LPI pattern is asserted after a link is established with the remote station. This period is defined as one second in the IEEE standard 802.3-az, version D2.0. The LPI LS timer is 10 bits wide, so the software can program up to 1023 ms.†

The LPI TW timer counts, in µs, the time expired since the deassertion of LPI. The terminal count of the timer is the value of resolved transmit TW that is the auto-negotiated time after which the MAC can resume the normal transmit operation. The LPI TW timer is 16 bits wide, so the software can program up to 65535 µs.†

The EMAC generates the LPI interrupt when the transmit or receive channel enters or exits the LPI state.†

Checksum Offload

Communication protocols such as TCP and UDP implement checksum fields, which help determine the integrity of data transmitted over a network. Because the most widespread use of Ethernet is to encapsulate TCP and UDP over IP datagrams, the EMAC has a Checksum Offload Engine (COE) to support checksum calculation and insertion in the transmit path, and error detection in the receive path. Supported offloading types:

- Transmit IP header checksum†
- Transmit TCP/UDP/ICMP checksum†
- Receive IP header checksum†
- Receive full checksum†

Frame Filtering

The EMAC implements the following types of filtering for receive frames.

Source Address or Destination Address Filtering

The Address Filtering Module checks the destination and source address field of each incoming packet.†

Unicast Destination Address Filter

Up to 128 MAC addresses for unicast perfect filtering are supported. The filter compares all 48 bits of the received unicast address with the programmed MAC address for any match. Default MacAddr0 is always enabled, other addresses MacAddr1–MacAddr127 are selected with an individual enable bit. For MacAddr1–MacAddr31 addresses, you can mask each byte during comparison with the corresponding received DA byte. This enables group address filtering for the DA. The MacAddr32-MacAddr127 addresses do not have mask control and all six bytes of the MAC address are compared with the received six bytes of DA.†

In hash filtering mode, the filter performs imperfect filtering for unicast addresses using a 256-bit hash table. It uses the upper ten bits of the CRC of the received destination address to index the content of the hash table. A value of 0 selects Bit 0 of the selected register, and a value of 111111 binary selects Bit 63 of the Hash Table register. If the corresponding bit is set to one, the unicast frame is said to have passed the hash filter; otherwise, the frame has failed the hash filter.†

Multicast Destination Address Filter

The MAC can be programmed to pass all multicast frames. In Perfect Filtering mode, the multicast address is compared with the programmed MAC Destination Address registers (1–31). Group address filtering is also supported. In hash filtering mode, the filter performs imperfect filtering using a 256-bit hash table. For hash filtering, it uses the upper ten bits of the CRC of the received multicast address to index the contents of the hash table. A value of 0 selects Bit 0 of the selected register and a value of 111111 binary selects Bit 63 of the Hash Table register. If the corresponding bit is set to one, then the multicast frame is said to have passed the hash filter; otherwise, the frame has failed the hash filter.†

Hash or Perfect Address Filter

The filter can be configured to pass a frame when its DA matches either the hash filter or the Perfect filter. This configuration applies to both unicast and multicast frames.†

Broadcast Address Filter

The filter does not filter any broadcast frames in the default mode. However, if the MAC is programmed to reject all broadcast frames, the filter drops any broadcast frame.†

Unicast Source Address Filter

The MAC can also perform a perfect filtering based on the source address field of the received frames. Group filtering with SA is also supported. You can filter a group of addresses by masking one or more bytes of the address.†

Inverse Filtering Operation (Invert the Filter Match Result at Final Output)

For both Destination and Source address filtering, there is an option to invert the filter-match result at the final output. The result of the unicast or multicast destination address filter is inverted in this mode.†

Destination and Source Address Filtering Summary

The tables below summarize the destination and source address filtering based on the type of frames received and the configuration of bits within the Mac_Frame_Filter register.†

Table 17-23: Destination Address Filtering†

Note: The "X" in the table represents a "don't care" term.

Frame Type	PR	HPF	HUC	DAIF	HMC	PM	DBF	Destination Address Filter Operation
Broadcast	1	X	X	X	X	X	X	Pass
	0	X	X	X	X	X	0	Pass
	0	X	X	X	X	X	1	Fail
Unicast	1	X	X	X	X	X	X	Pass all frames
	0	X	0	0	X	X	X	Pass on Perfect/Group filter match
	0	X	0	1	X	X	X	Fail on Perfrect/Group filter match
	0	0	1	0	X	X	X	Pass on Hash filter match
	0	0	1	1	X	X	X	Fail on Hash filter match
	0	1	1	0	X	X	X	Pass on Hash or Perfect/Group filter match
	0	1	1	1	X	X	X	Fail on Hash or Perfect/Group filter match
Multicast	1	X	X	X	X	X	X	Pass all frames
	X	X	X	X	X	1	X	Pass all frames
	0	X	X	0	0	0	X	Pass on Perfect/Group filter match and drop Pause frames if PCF= 0X
	0	0	X	0	1	0	X	Pass on Hash filter match and drop Pause frames if PCF = 0X
	0	1	X	0	1	0	X	Pass on Hash or Perfect/Group filter match and drop Pause frames if PCF = 0X
	0	X	X	1	0	0	X	Fail on Perfect/Group filter match and drop Pause frames if PCF=0X
	0	0	X	1	1	0	X	Fail on Hash filter match and drop Paus frames if PCF = 0X
	0	1	X	1	1	0	X	Fail Hash on Perfect/Group filter match and drop Pause frames if PCF = 0X

Table 17-24: Source Address Filtering†

Frame Type	PR	DAIF	DBF	Source Address Filter Operation
Unicast	1	X	X	Pass all frames
	0	0	0	Pass status on Perfect or Group filter match but do not drop frames that fail.
	0	1	0	Fail on Perfect or Group filter match but do not drop frame
	0	0	1	Pass on Perfect or Group filter match and drop frames that fail
	0	1	1	Fail on Perfect or Group filter match and drop frames that fail

VLAN Filtering

The EMAC supports the two kinds of VLAN filtering:

- VLAN tag-based filtering†
- VLAN hash filtering†

VLAN Tag-Based Filtering

In the VLAN tag-based frame filtering, the MAC compares the VLAN tag of the received frame and provides the VLAN frame status to the application. Based on the programmed mode, the MAC compares the lower 12 bits or all 16 bits of the received VLAN tag to determine the perfect match. If VLAN tag filtering is enabled, the MAC forwards the VLAN-tagged frames along with VLAN tag match status and drops the VLAN frames that do not match. You can also enable the inverse matching for VLAN frames. In addition, you can enable matching of SVLAN tagged frames along with the default Customer Virtual Local Area Network (C-VLAN) tagged frames.†

VLAN Hash Filtering with a 16-Bit Hash Table

The MAC provides VLAN hash filtering with a 16-bit hash table. The MAC also supports the inverse matching of the VLAN frames. In inverse matching mode, when the VLAN tag of a frame matches the perfect or hash filter, the packet should be dropped. If the VLAN perfect and VLAN hash match are enabled, a frame is considered as matched if either the VLAN hash or the VLAN perfect filter matches. When inverse match is set, a packet is forwarded only when both perfect and hash filters indicate mismatch.†

Layer 3 and Layer 4 Filters

Layer 3 filtering refers to source address and destination address filtering. Layer 4 filtering refers to source port and destination port filtering. The frames are filtered in the following ways:†

- Matched frames†
- Unmatched frames†
- Non-TCP or UDP IP frames†

Matched Frames

The MAC forwards the frames, which match all enabled fields, to the application along with the status. The MAC gives the matched field status only if one of the following conditions is true:†

- All enabled Layer 3 and Layer 4 fields match.†
- At least one of the enabled field matches and other fields are bypassed or disabled.†

Using the CSR set, you can define up to four filters, identified as filter 0 through filter 3. When multiple Layer 3 and Layer 4 filters are enabled, any filter match is considered as a match. If more than one filter matches, the MAC provides status of the lowest filter with filter 0 being the lowest and filter 3 being the highest. For example, if filter 0 and filter 1 match, the MAC gives the status corresponding to filter 0.†

Unmatched Frames

The MAC drops the frames that do not match any of the enabled fields. You can use the inverse match feature to block or drop a frame with specific TCP or UDP over IP fields and forward all other frames. You can configure the EMAC so that when a frame is dropped, it receives a partial frame with appropriate abort status or drops it completely.†

NonTCP or UDP IP Frames

By default, all non-TCP or UDP IP frames are bypassed from the Layer 3 and Layer 4 filters. You can optionally program the MAC to drop all non-TCP or UDP over IP frames.†

Layer 3 and Layer 4 Filters Register Set

The MAC implements a set of registers for Layer 3 and Layer 4 based frame filtering. In this register set, there is a control register for frame filtering and five address registers.†

You can configure the MAC to have up to four such independent set of registers.†

The registers available for programming are as follows:

- `gmacgrp_l3_l4_control0` through `gmacgrp_l3_l4_control3` registers: Layer and Layer 4 Control registers
- `gmacgrp_layer4_address0` through `gmacgrp_layer4_address3` registers: Layer 4 Address registers
- `gmacgrp_layer3_addr0_reg0` through `gmacgrp_layer3_addr0_reg3` registers: Layer 3 Address 0 registers
- `gmacgrp_layer3_addr1_reg0` through `gmacgrp_layer3_addr1_reg3` registers: Layer 3 Address 1 registers
- `gmacgrp_layer3_addr2_reg0` through `gmacgrp_layer3_addr2_reg3` registers: Layer 3 Address 2 registers
- `gmacgrp_layer3_addr3_reg0` through `gmacgrp_layer3_addr3_reg3` registers: Layer 3 Address 3 registers

Related Information

Ethernet MAC Address Map and Register Definitions on page 17-78

Layer 3 Filtering

The EMAC supports perfect matching or inverse matching for the IP Source Address and Destination Address. In addition, you can match the complete IP address or mask the lower bits. †

For IPv6 frames filtering, you can enable the last four data registers of a register set to contain the 128-bit IP Source Address or IP Destination Address. The IP Source or Destination Address should be programmed in the order defined in the IPv6 specification. The specification requires that you program

the first byte of the received frame IP Source or Destination Address in the higher byte of the register. Subsequent registers should follow the same order.†

For IPv4 frames filtering, you can enable the second and third data registers of a register set to contain the 32-bit IP Source Address and IP Destination Address. The remaining two data registers are reserved. The IP Source and Destination Address should be programmed in the order defined in the IPv4 specification. The specification requires that you program the first byte of received frame IP Source and Destination Address in the higher byte of the respective register.†

Layer 4 Filtering

The EMAC supports perfect matching or inverse matching for TCP or UDP Source and Destination Port numbers. However, you can program only one type (TCP or UDP) at a time. The first data register contains the 16-bit Source and Destination Port numbers of TCP or UDP, that is, the lower 16 bits for Source Port number and higher 16 bits for Destination Port number. †

The TCP or UDP Source and Destination Port numbers should be programmed in the order defined in the TCP or UDP specification, that is, the first byte of TCP or UDP Source and Destination Port number in the received frame is in the higher byte of the register.†

Clocks and Resets

Clock Structure

The Ethernet Controller has four main clock domains.

- l4_mp_clk clock
- EMAC RX clock
- EMAC TX clock
- clk_ptp_ref

Figure 17-15: EMAC Clock Diagram

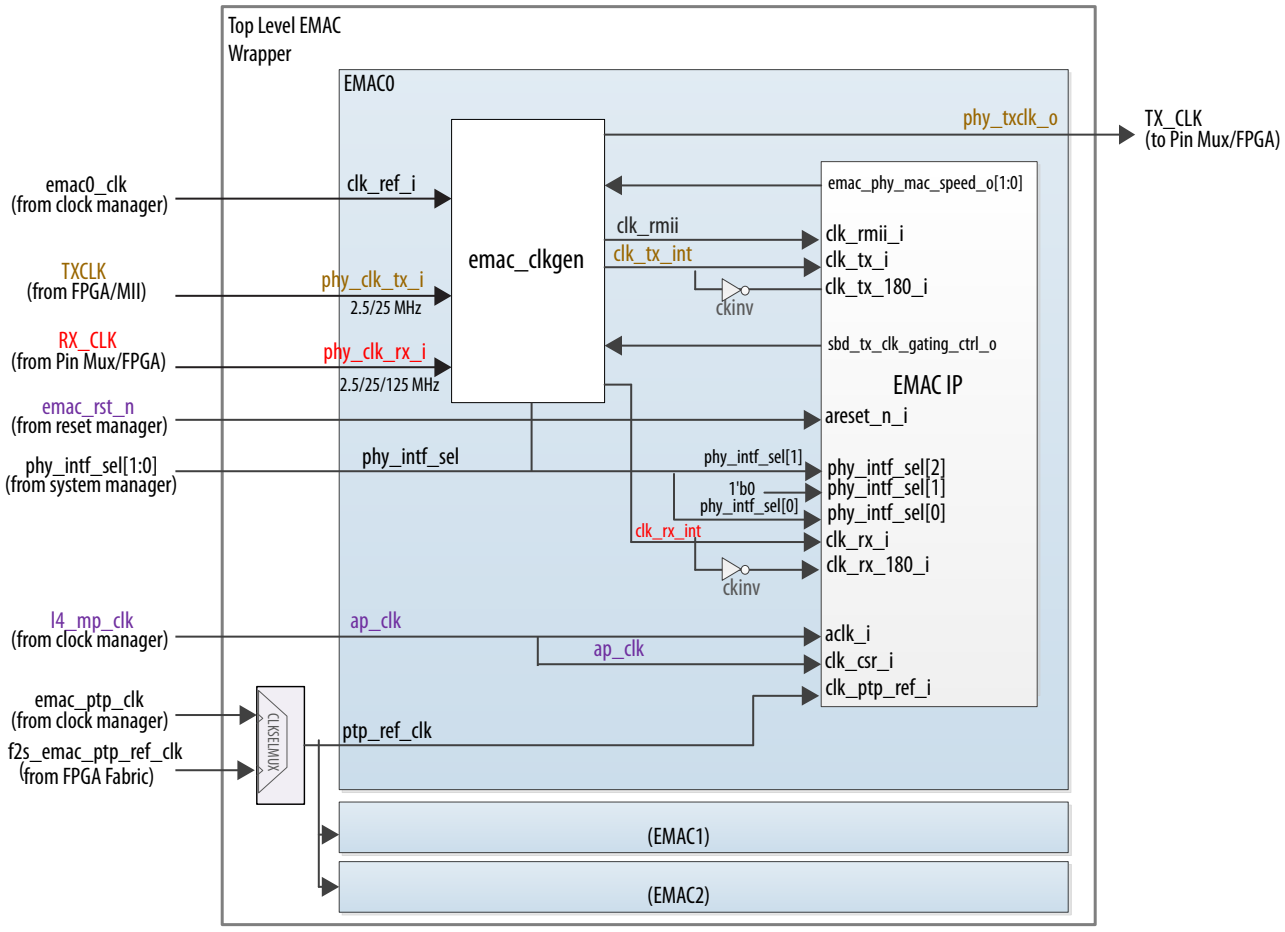
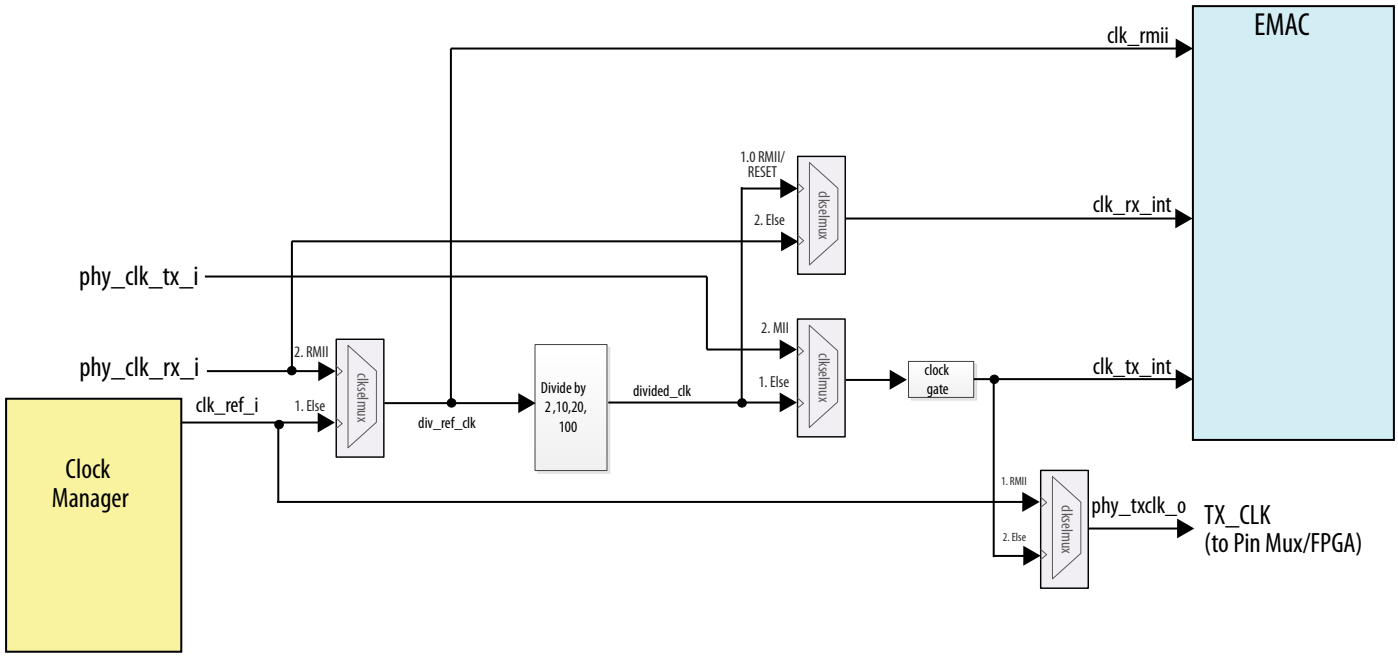


Figure 17-16: emac_clkgen Module



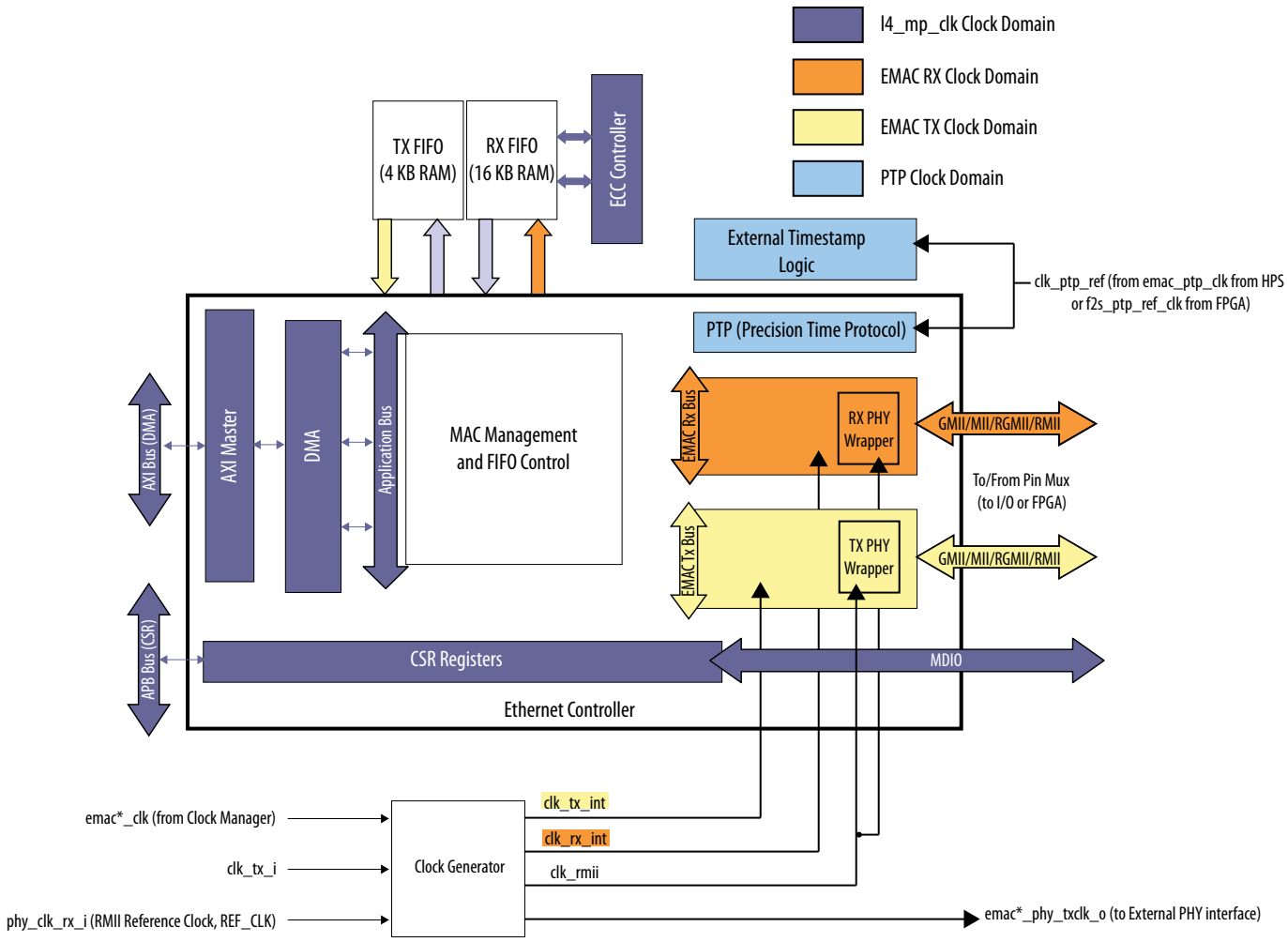
Depending on the interface, different clock domains are used:

- When the DMA master interface is used for EMAC packet transfers, the `l4_mp_clk` is used as a clock source for both the AXI bus and the CSR register interface. This clock domain is a fully synchronous.
- The RX and TX FIFO RAMs are driven by the `l4_mp_clk`.
- The MDIO interface's clock domain is a derivative of the CSR clock, which comes from l4_mp_clk. Typically MDC clock has a frequency between 1 to 2.5 MHz, however, faster MDC frequencies are supported in this design.
- The EMAC contains an RX datapath, TX datapath and timestamp interface that all run on separate clock domains.
  - The RX datapath is in the EMAC RX clock domain.
  - The TX datapath is in the EMAC TX clock domain.
  - The timestamp interface is in the clk_ptp_ref clock domain.

The timestamp clock domain provides the capability for EMAC0 to be a timestamp master with internal timestamp enabled and the other two EMACs to be timestamp slaves using the timestamp generated from EMAC0.

The diagram below summarizes the clock domains of the EMAC module:

Figure 17-17: EMAC Clock Domains



The following table summarizes the clock inputs and outputs to the EMAC.

Table 17-25: EMAC Module Clock Inputs and Outputs

Clock	Input/Output	Frequency	Source	Description
`l4_mp_clk`	Input	200 MHz	Clock Manager	Application clock for DMA bus interface, CSR interface and ECC FIFO RAMs.

Clock	Input/Output	Frequency	Source	Description
clk_ptp_ref	Input	up to 100 MHz	Clock Manager or FPGA fabric	This signal is sourced by either the PTP reference clock from the Clock Manager or the FPGA fabric. The source can be selected through the ptp_clk_sel bit of the emac_global register in the System Manager module. When the bit is clear, the emac_ptp_clk is selected and when it is set, the f2s_ptp_ref_clk is selected.
emac*_clk	Input	Variable depending on divider value of programmed in Clock Manager.	Input from Clock Manager	This signal is configured in the Clock Manager module and can be enabled to drive the clk_tx_in and clk_rx int signals to the TX and RX clock domains.
clk_tx_i	Input	Used only in MII mode as a 25 or 2.5 MHz clock source at 100 Mbps and 10 Mbps, respectively.	Input from FPGA fabric I/O	This signal is used only in MII mode as a TX reference clock. **Note:** This clock must be able to perform glitch free switching between 2.5 and 25 MHz.
phy_clk_rx_i	Input	• GMII mode: 125 MHz • RGMII mode: 125, 25, or 2.5 MHz • MII mode: 25 or 2.5 MHz • RMII mode: 50 MHz	This clock input is driven to FPGA or by an HPS I/O input from an external PHY	For all modes except, RMII, this clock signal is the RX PHY input clock. For RMII mode, this input is a 50 MHz reference clock (REF_CLK) from the board or from phy_txclk_o that is divided down automatically to generate the datapath clocks, emac*_clk_rx_i and emac*_clk_tx_i signals. These datapath clocks are 2.5 MHz when operating in 10 Mbps mode and 25 MHz when operating in 100 Mbps mode.
phy_txclk_o	Output	125, 50, 25, or 2.5 MHz	From internal HPS clk_tx_int to HPS I/O or from FPGA fabric.	This signal is an TX output clock to the PHY. In RMII mode, this signal can provide the reference clock (50 MHz in 100M /10 Mbps).

Clock Gating for EEE

For the RGMII PHY interface, you can gate the transmit clock for Energy Efficient Ethernet (EEE) applications.

Related Information

Programming Guidelines for Energy Efficient Ethernet on page 17-75

Reset

The EMAC module accepts a single reset input, `emac_rst_n`, which is active low.

Note: In all modes, the EMAC core depends on the PHY clocks to be active for the internal EMAC clock sources to be valid.

Taking the Ethernet MAC Out of Reset

When a cold or warm reset is issued in the HPS, the Reset Manager resets the EMAC module and holds it in reset until software releases it.

After the MPU boots up, it can deassert the reset signal by clearing the appropriate bits in the Reset Manager's corresponding reset register. Before deasserting the reset signal, you must make sure the PHY interface type and all other corresponding EMAC settings in the System Manager have been configured. For details about reset registers, refer to the "Module Reset Signals" section in the *Reset Manager* chapter. For more information about EMAC configuration in the System Manager, refer to the "System Level EMAC Configuration Registers" section.

EMAC ECC RAM Reset

An EMAC ECC RAM reset asserts a reset to both the memory and the multiplexed EMAC bus interface clock, `ap_clk`. You should ensure that both the EMAC ECC RAM and the EMAC Module resets are deasserted before beginning transactions. Program the `emac*ocp` bits and the `emac*` bits in the `per0modrst` register of the Reset Manager to deassert reset in the EMAC's ECC RAM and the EMAC module, respectively.

Related Information

- **System Level EMAC Configuration Registers** on page 17-69
- **Module Reset Signals** on page 3-8
  For more information about reset registers refer to this section in the *Reset Manager* chapter.

Interrupts

Interrupts are generated as a result of specific events in the EMAC and external PHY device. The interrupt status register indicates all conditions which may trigger an interrupt and the interrupt enable register determines which interrupts can propagate.

Ethernet MAC Programming Model

The initialization and configuration of the EMAC and its interface is a multi-step process that includes system register programming in the System Manager and Clock Manager and configuration of clocks in multiple domains.

Note: When the EMAC interfaces to HPS I/O and register content is being transferred to a different clock domain after a write operation, no further writes should occur to the same location until the first

write is updated. Otherwise, the second write operation does not get updated to the destination clock domain. Thus, the delay between two writes to the same register location should be at least 4 cycles of the destination clock (PHY receive clock, PHY transmit clock, or PTP clock).† If the CSR is accessed multiple times quickly, you must ensure that a minimum number of destination clock cycles have occurred between accesses.

Note: If the EMAC signals are routed through the FPGA fabric and it is assumed that the transmit clock supplied by the FPGA fabric switches within 6 transmit clock cycles, then the minimum time required between two write accesses to the same register is 10 transmit clock cycles.

System Level EMAC Configuration Registers

In addition to the registers in the Ethernet Controller, there are other system level registers in the Clock Manager, System Manager and Reset Manager that must be programmed in order to configure the EMAC and its interfaces.

The following table gives a summary of the important System Manager clock register bits that control operation of the EMAC. These register bits are static signals that must be set while the corresponding EMAC is in reset.

Table 17-26: System Manager Clock and Interface Settings

Register.Field	Description
`emac_global.ptp_clk_sel`	1588 PTP reference clock. This bit selects the source of the 1588 PTP reference clock. • 0x0= `emac_ptp_clk` (default from Clock Manager) • 0x1=`f2s_emac_ptp_ref_clk` (from FPGA fabric; in this case, the FPGA must be in usermode with an active reference clock)

`emac0.phy_intf_sel`
`emac1.phy_intf_sel`
`emac2.phy_intf_sel`
	PHY Interface Select. These two bits set the PHY mode.
	• 0x0= GMII or MII
	• 0x1= RGMII
	• 0x2= RMII
	• 0x3= RESET (default)

The following table summarizes the important System Manager configuration register bits. All of the fields, except the AXI cache settings, are assumed to be static and must be set before the EMAC is brought out of reset. If the FPGA interface is used, the FPGA must be in user mode and enabled with the appropriate clock signals active before the EMAC can be brought out of reset.

Table 17-27: System Manager Static Control Settings

Register.Field	Description
fpgaintf_en_3.emac0 fpgaintf_en_3.emac1 fpgaintf_en_3.emac2	FPGA interface to EMAC disable. This field is used to disable signals from the FPGA to the EMAC modules that could potentially interfere with the EMAC's or FPGA's operation. • 0x0= Disable (default) • 0x1=Enable
emac0.axi_disable emac1.axi_disable emac2.axi_disable	AXI Disable. Disables the AXI bus to EMAC. • 0x0= Enable (default) • 0x1= Disable
emac0.awcache emac1.awcache emac2.awcache emac0.arcache emac1.arcache emac2.arcache	EMAC AXI Master AxCACHE settings. It is recommended that these bits are set while the EMAC is idle or in reset.
emac0.awprot emac1.awprot emac2.awprot emac0.arprot emac1.arprot emac2.arprot	EMAC Master AxPROT settings. It is recommended that these bits are set while the EMAC is idle or in reset.
emac0.ptp_ref_sel emac1.ptp_ref_sel emac2.ptp_ref_sel	Internal/External Timestamp reference. This field selects if the timestamp reference is internally or externally generated. EMAC0 may be the master to generate the timestamp for EMAC1 and EMAC2. EMAC0 must be set to internal timestamp; EMAC1 and EMAC2 may be set either to internal or external. • 0x0= Internal (default) • 0x1= External

Various registers within the Clock Manager must also be configured in order for the EMAC controller to perform properly.

Table 17-28: Clock Manager Settings

Register.Field	Description
en.emacptpen	emac_ptp_clk output enable.
en.emac0en en.emac1en en.emac2en	Enables clock emac0_clk, emac1_clk and emac2_clk output. **Note:** There are corresponding ens and enr registers that allow the same fields to be set or cleared on a bit-by-bit basis.
bypass.emacptp	EMAC PTP clock bypass. This bit indicates if the emac_ptp_clk is bypassed to the input clock reference of the peripheral PLL. • 0x0= No bypass occurs • 0x1= emac_ptp_clk is bypassed to the input clock reference of the main PLL. **Note:** There are corresponding bypasss and bypassr registers that allow the same bits to be set or cleared on a bit-by-bit basis.
bypass.emaca bypass.emacb	Clock Bypass. This bit indicates whether emaca_free_clk or emacb_free_clk is bypassed to the input clock reference of the main PLL. • 0x0= No bypass occurs • 0x1= emac*_free_clk is bypassed to the input clock reference of the main PLL. **Note:** There are corresponding bypasss and bypassr registers that allow the same bits to be set or cleared on a bit-by-bit basis.
emacctl.emac0sel emacctl.emac1sel emacctl.emac2sel	EMAC clock source select. This bit selects the source for the emac*clk as either emaca_free_clk or emacb_free clk • 0x0= emaca_free_clk • 0x1=emacb_free_clk

EMAC FPGA Interface Initialization

To initialize the Ethernet controller to use the FPGA GMII/MII interface, specific software steps must be followed.

In general, the FPGA interface must be active in user mode with valid PHY clocks, the Ethernet Controller must be in a reset state during static configuration and the clock must be active and valid before the Ethernet Controller is brought out of reset.

1. After the HPS is released from cold or warm reset, reset the Ethernet Controller module by setting the appropriate `emac*` bit in the `per0modrst` register in the Reset Manager.
2. Configure the EMAC Controller clock to 250 MHz by programming the appropriate registers in the Clock Manager.
3. Bring the Ethernet PHY out of reset to verify that there are RX PHY clocks.
4. If the PTP clock source is from the FPGA, ensure that the FPGA `f2s_ptp_ref_clk` is active.
5. The soft GMII/MII adaptor must be loaded with active clocks propagating. The FPGA must be configured to user mode and a reset to the user soft FPGA IP may be required to propagate the PHY clocks to the HPS.
6. Once all clock sources are valid, apply the following clock settings:
   a. Program the `phy_intf_sel` field of the `emac*` register in the System Manager to 0x0 to select GMII/MII PHY interface.
   b. If the PTP clock source is from the FPGA, set the `ptp_clk_sel` bit to 0x1 in the `emac_global` register of the System Manager.
   c. Enable the Ethernet Controller FPGA interface by setting the `emac_*` bit in the `fpgaintf_en_3` register of the System Manager.
7. Configure all of the EMAC static settings if the user requires a different setting from the default value. These settings include the AxPROT[1:0] and AxCACHE signal values which are programmed in the `emac*` register of the System Manager.
8. Execute a register read back to confirm the clock and static configuration settings are valid.
9. After confirming the settings are valid, software can clear the `emac*` bit in the `per0modrst` register of the Reset Manager to bring the EMAC out of reset..

When these steps are completed, general Ethernet controller and DMA software initialization and configuration can continue.

Note: These same steps can be applied to convert the HPS GMII to an RGMII, RMII or SGMII interface through the FPGA, except that in step 5 during FPGA configuration, you would load the appropriate soft adaptor for the interface and apply reset to it as well. The PHY interface select encoding would remain as 0x0. For the SGMII interface additional external transceiver logic would be required. Routing the Ethernet signals through the FPGA is useful for designs that are pin-limited in the HPS.

EMAC HPS Interface Initialization

To initialize the Ethernet controller to use the HPS interface, specific software steps must be followed including selecting the correct PHY interface through the System Manager.

In general, the Ethernet Controller must be in a reset state during static configuration and the clock must be active and valid before the Ethernet Controller is brought out of reset.

1. After the HPS is released from cold or warm reset, reset the Ethernet Controller module by setting the appropriate `emac*` bit in the `per0modrst` register in the Reset Manager.
2. Configure the EMAC Controller clock to 250 MHz by programming the appropriate registers in the Clock Manager.
3. Bring the Ethernet PHY out of reset to verify that there are RX PHY clocks.
4. When all the clocks are valid, program the following clock settings:

   a. Program the `phy_intf_sel` field of the `emac*` register in the System Manager to 0x1 or 0x2 to select RGMII or RMII PHY interface.
   b. Disable the Ethernet Controller FPGA interface by clearing the `emac_*` bit in the `fpgaintf_en_3` register of the System Manager.
5. Configure all of the EMAC static settings if the user requires a different setting from the default value. These settings include the AxPROT[1:0] and AxCACHE signal values, which are programmed in the `emac*` register of the System Manager.
6. Execute a register read back to confirm the clock and static configuration settings are valid.
7. After confirming the settings are valid, software can clear the `emac*` bit in the `per0modrst` register of the Reset Manager to bring the EMAC out of reset..

When these steps are completed, general Ethernet controller and DMA software initialization and configuration can continue.

DMA Initialization

This section provides the instructions for initializing the DMA registers in the proper sequence. This initialization sequence can be done after the EMAC interface initialization has been completed. Perform the following steps to initialize the DMA:

1. Provide a software reset to reset all of the EMAC internal registers and logic. (DMA Register 0 (Bus Mode Register) – bit 0). †
2. Wait for the completion of the reset process (poll bit 0 of the DMA Register 0 (Bus Mode Register), which is only cleared after the reset operation is completed). †
3. Poll the bits of Register 11 (AXI Status) to confirm that all previously initiated (before software reset) or ongoing transactions are complete.

   Note: If the application cannot poll the register after soft reset (because of performance reasons), then it is recommended that you continue with the next steps and check this register again (as mentioned in step **12** on page 1-74) before triggering the DMA operations.†
4. Program the following fields to initialize the Bus Mode Register by setting values in DMA Register 0 (Bus Mode Register):†
   - Mixed Burst and AAL
   - Fixed burst or undefined burst†
   - Burst length values and burst mode values†
   - Descriptor Length (only valid if Ring Mode is used)†
5. Program the interface options in Register 10 (AXI Bus Mode Register). If fixed burst-length is enabled, then select the maximum burst-length possible on the bus (bits[7:1]).†
6. Create a proper descriptor chain for transmit and receive. In addition, ensure that the receive descriptors are owned by DMA (bit 31 of descriptor should be set). When OSF mode is used, at least two descriptors are required.
7. Make sure that your software creates three or more different transmit or receive descriptors in the chain before reusing any of the descriptors.†
8. Initialize receive and transmit descriptor list address with the base address of the transmit and receive descriptor (Register 3 (Receive Descriptor List Address Register) and Register 4 (Transmit Descriptor List Address Register) respectively).†
9. Program the following fields to initialize the mode of operation in Register 6 (Operation Mode Register):

- Receive and Transmit Store And Forward†
- Receive and Transmit Threshold Control (RTC and TTC)†
- Hardware Flow Control enable†
- Flow Control Activation and De-activation thresholds for MTL Receive and Transmit FIFO buffers (RFA and RFD)†
- Error frame and undersized good frame forwarding enable†
- OSF Mode†

10. Clear the interrupt requests, by writing to those bits of the status register (interrupt bits only) that are set. For example, by writing 1 into bit 16, the normal interrupt summary clears this bit (DMA Register 5 (Status Register)).†
11. Enable the interrupts by programming Register 7 (Interrupt Enable Register).†

    Note: Perform step 12 on page 1-74 only if you did not perform step 3 on page 1-73.†
12. Read Register 11 (AHB or AXI Status) to confirm that all previous transactions are complete.†

    Note: If any previous transaction is still in progress when you read the Register 11 (AXI Status), then it is strongly recommended to check the slave components addressed by the master interface.†
13. Start the receive and transmit DMA by setting SR (bit 1) and ST (bit 13) of the control register (DMA Register 6 (Operation Mode Register). †

EMAC Initialization and Configuration

The following EMAC configuration operations can be performed after DMA initialization. If the EMAC initialization and configuration is done before the DMA is set up, then enable the MAC receiver (last step below) only after the DMA is active. Otherwise, the received frame could fill the RX FIFO buffer and overflow.

1. Program the `GMII Address Register` (offset 0x10) for controlling the management cycles for the external PHY. Bits[15:11] of the `GMII Address Register` are written with the Physical Layer Address of the PHY before reading or writing. Bit 0 indicates if the PHY is busy and is set before reading or writing to the PHY management interface. †
2. Read the 16-bit data of the `GMII Data Register` from the PHY for link up, speed of operation, and mode of operation, by specifying the appropriate address value in bits[15:11] of the `GMII Address Register`. †
3. Provide the MAC address registers (`MAC Address0 High Register` through `MAC Address15 High Register` and `MAC Address0 Low Register` through `MAC Address15 Low Register`).
4. Program the `Hash Table Registers` 0 through 7 (offset 0x500 to 0x51C).
5. Program the following fields to set the appropriate filters for the incoming frames in the `MAC Frame Filter Register`: †

    - Receive All †
    - Promiscuous mode †
    - Hash or Perfect Filter †
    - Unicast, multicast, broadcast, and control frames filter settings †
6. Program the following fields for proper flow control in the `Flow Control Register`: †

- Pause time and other pause frame control bits †
- Receive and Transmit Flow control bits †
- Flow Control Busy/Backpressure Activate †

7. Program the `Interrupt Mask Register` bits, as required and if applicable for your configuration. †
8. Program the appropriate fields in `MAC Configuration Register` to configure receive and transmit operation modes. After basic configuration is written, set bit 3 (`TE`) and bit 2 (`RE`) in this register to enable the receive and transmit state machines. †

Note: Do not change the configuration (such as duplex mode, speed, port, or loopback) when the EMAC DMA is actively transmitting or receiving. Software should change these parameters only when the EMAC DMA transmitter and receiver are not active.

Performing Normal Receive and Transmit Operation

For normal operation, perform the following steps: †

1. For normal transmit and receive interrupts, read the interrupt status. Then, poll the descriptors, reading the status of the descriptor owned by the Host (either transmit or receive). †
2. Set appropriate values for the descriptors, ensuring that transmit and receive descriptors are owned by the DMA to resume the transmission and reception of data. †
3. If the descriptors are not owned by the DMA (or no descriptor is available), the DMA goes into SUSPEND state. The transmission or reception can be resumed by freeing the descriptors and issuing a poll demand by writing 0 into the TX/RX poll demand registers, (Register 1 (Transmit Poll Demand Register) and Register 2 (Receive Poll Demand Register)). †
4. The values of the current host transmitter or receiver descriptor address pointer can be read for the debug process (Register 18 (Current Host Transmit Descriptor Register) and Register 19 (Current Host Receive Descriptor Register)). †
5. The values of the current host transmit buffer address pointer and receive buffer address pointer can be read for the debug process (Register 20 (Current Host Transmit Buffer Address Register) and Register 21 (Current Host Receive Buffer Address Register)). †

Stopping and Starting Transmission

Perform the following steps to pause the transmission for some time: †

1. Disable the transmit DMA (if applicable), by clearing bit 13 (Start or Stop Transmission Command) of Register 6 (Operation Mode Register). †
2. Wait for any previous frame transmissions to complete. You can check this by reading the appropriate bits of Register 9 (Debug Register). †
3. Disable the EMAC transmitter and EMAC receiver by clearing Bit 3 (TE) and Bit 2 (RE) in Register 0 (MAC Configuration Register). †
4. Disable the receive DMA (if applicable), after making sure that the data in the RX FIFO buffer is transferred to the system memory (by reading Register 9 (Debug Register). †
5. Make sure that both the TX FIFO buffer and RX FIFO buffer are empty. †
6. To re-start the operation, first start the DMA and then enable the EMAC transmitter and receiver. †

Programming Guidelines for Energy Efficient Ethernet

Entering and Exiting the TX LPI Mode

The Energy Efficient Ethernet (EEE) feature is available in the EMAC. To use it, perform the following steps during EMAC initialization:

1. Read the PHY register through the MDIO interface, check if the remote end has the EEE capability, and then negotiate the timer values. †
2. Program the PHY registers through the MDIO interface (including the RX_CLK_stoppable bit that indicates to the PHY whether to stop the RX clock in LPI mode.) †
3. Program Bits[16:5], LST, and Bits[15:0], TWT, in Register 13 (LPI Timers Control Register). †
4. Read the link status of the PHY chip by using the MDIO interface and update Bit 17 (PLS) of Register 12 (LPI Control and Status Register) accordingly. This update should be done whenever the link status in the PHY changes. †
5. Set Bit 16 (LPIEN) of Register 12 (LPI Control and Status Register) to make the MAC enter the LPI state. The MAC enters the LPI mode after completing the transmission in progress and sets Bit 0 (TLPIEN). †

   Note: To make the MAC enter the LPI state only after it completes the transmission of all queued frames in the TX FIFO buffer, you should set Bit 19 (LPITXA) in Register 12 (LPI Control and Status Register). †

   Note: To switch off the transmit clock during the LPI state, use the `sbd_tx_clk_gating_ctrl_o` signal for gating the clock input. †

   Note: To switch off the CSR clock or power to the rest of the system during the LPI state, you should wait for the TLPIEN interrupt of Register 12 (LPI Control and Status Register) to be generated. Restore the clocks before performing step 6 on page 1-76 when you want to come out of the LPI state. †
6. Clear Bit 16 (LPIEN) of Register 12 (LPI Control and Status Register) to bring the MAC out of the LPI state. †

The MAC waits for the time programmed in Bits [15:0], TWT, before setting the TLPIEX interrupt status bit and resuming the transmission. †

Gating Off the CSR Clock in the LPI Mode

You can gate off the CSR clock to save the power when the MAC is in the Low-Power Idle (LPI) mode.†

Gating Off the CSR Clock in the RX LPI Mode

The following operations are performed when the MAC receives the LPI pattern from the PHY. †

1. The MAC RX enters the LPI mode and the RX LPI entry interrupt status [RLPIEN interrupt of Register 12 (`LPI_Control_Status`)] is set. †
2. The interrupt pin (`sbd_intr_o`) is asserted. The `sbd_intr_o` interrupt is cleared when the host reads the Register 12 (LPI_Control_Status). †

After the `sbd_intr_o` interrupt is asserted and the MAC TX is also in the LPI mode, you can gate off the CSR clock. If the MAC TX is not in the LPI mode when you gate off the CSR clock, the events on the MAC transmitter do not get reported or updated in the CSR. †

For restoring the CSR clock, wait for the LPI exit indication from the PHY after which the MAC asserts the LPI exit interrupt on `lpi_intr_o` (synchronous to `clk_rx_i`). The `lpi_intr_o` interrupt is cleared when Register 12 is read. †

Gating Off the CSR Clock in the TX LPI Mode

The following operations are performed when Bit 16 (LPIEN) of Register 12 (LPI Control and Status Register) is set: †

1. The Transmit LPI Entry interrupt (TLPIEN bit of Register 12) is set. †
2. The interrupt pin (`sbd_intr_o`) is asserted. The `sbd_intr_o` interrupt is cleared when the host reads the Register 12. †

After the `sbd_intr_o` interrupt is asserted and the MAC RX is also in the LPI mode, you can gate off the CSR clock. If the MAC RX is not in the LPI mode when you gate off the CSR clock, the events on the MAC receiver do not get reported or updated in the CSR. †

To restore the CSR clock, switch on the CSR clock when the MAC has to come out of the TX LPI mode. †

After the CSR clock is resumed, clear Bit 16 (LPIEN) of Register 12 (LPI Control and Status Register) to bring the MAC out of the LPI mode. †

Programming Guidelines for Flexible Pulse-Per-Second (PPS) Output

Generating a Single Pulse on PPS

To generate single Pulse on PPS: †

1. Program 11 or 10 (for interrupt) in Bits [6:5], TRGTMODSEL, of Register 459 (PPS Control Register) to instruct the MAC to use the Target Time registers (register 455 and 456) for the start time of PPS signal output. †
2. Program the start time value in the Target Time registers (register 455 and 456). †
3. Program the width of the PPS signal output in Register 473 (PPS0 Width Register). †
4. Program Bits [3:0], PPSCMD, of Register 459 (PPS Control Register) to 0001 to instruct the MAC to generate a single pulse on the PPS signal output at the time programmed in the Target Time registers (register 455 and 456). †

Once the PPSCMD is executed (PPSCMD bits = 0), you can cancel the pulse generation by giving the Cancel Start Command (PPSCMD=0011) before the programmed start time elapses. You can also program the behavior of the next pulse in advance. To program the next pulse: †

1. Program the start time for the next pulse in the Target Time registers (register 455 and 456). This time should be more than the time at which the falling edge occurs for the previous pulse. †
2. Program the width of the next PPS signal output in Register 473 (PPS0 Width Register). †
3. Program Bits [3:0], PPSCMD, of Register 459 (PPS Control Register) to generate a single pulse on the PPS signal output after the time at which the previous pulse is de-asserted. And at the time programmed in Target Time registers. If you give this command before the previous pulse becomes low, then the new command overwrites the previous command and the EMAC may generate only 1 extended pulse.

Generating a Pulse Train on PPS

To generate a pulse train on PPS: †

1. Program 11 or 10 (for an interrupt) in Bits [6:5], TRGTMODSEL, of Register 459 (PPS Control Register) to instruct the MAC to use the Target Time registers (register 455 and 456) for the start time of the PPS signal output. †
2. Program the start time value in the Target Time registers (register 455 and 456). †
3. Program the interval value between the train of pulses on the PPS signal output in Register 473 (PPS0 Width Register). †
4. Program the width of the PPS signal output in Register 473 (PPS0 Width Register). †
5. Program Bits[3:0], PPSCMD, of Register 459 (PPS Control Register) to 0010 to instruct the MAC to generate a train of pulses on the PPS signal output with the start time programmed in the Target Time registers (register 455 and 456). By default, the PPS pulse train is free-running unless stopped by 'STOP Pulse train at time' or 'STOP Pulse Train immediately' commands. †
6. Program the stop value in the Target Time registers (register 455 and 456). Ensure that Bit 31 (TSTRBUSY) of Register 456 (Target Time Nanoseconds Register) is clear before programming the Target Time registers (register 455 and 456) again. †
7. Program the PPSCMD field (bit 3:0) of Register 459 (PPS Control Register) to 0100 to stop the train of pulses on the PPS signal output after the programmed stop time specified in step 6 on page 1-78 elapses. †

You can stop the pulse train at any time by programming 0101 in the PPSCMD field. Similarly, you can cancel the Stop Pulse train command (given in step 7 on page 1-78) by programming 0110 in the PPSCMD field before the time (programmed in step 6 on page 1-78) elapses. You can cancel the pulse train generation by programming 0011 in the PPSCMD field before the programmed start time (in step 2 on page 1-78) elapses. †

Generating an Interrupt without Affecting the PPS

Bits [6:5], TRGTMODSEL, of Register 459 (PPS Control Register) enable you to program the Target Time registers (register 455 and 456) to do any one of the following: †

- Generate only interrupts. †
- Generate interrupts and the PPS start and stop time. †
- Generate only PPS start and stop time. †

To program the Target Time registers (register 455 and 456) to generate only interrupt events: †

1. Program 00 (for interrupt) in Bits [6:5], TRGTMODSEL, of Register 459 (PPS Control Register) to instruct the MAC to use the Target Time registers (register 455 and 456) for the target time interrupt. †
2. Program a target time value in the Target Time registers (register 455 and 456) to instruct the MAC to generate an interrupt when the target time elapses. If Bits [6:5], TRGTMODSEL, are changed (for example, to control the PPS), then the interrupt generation is overwritten with the new mode and new programmed Target Time register value.

Ethernet MAC Address Map and Register Definitions

The address map and register definitions pertain to the following modules:

- EMAC Module 0
- EMAC Module 1
- EMAC Module 2

In addition to the EMAC modules, the address map and register definitions for the ECC controllers of the 4-KB Tx FIFO RAM and the 16-KB Rx FIFO RAM are listed in this section.

Related Information

Error Checking and Correction Controller on page 11-1
For more information on the functionality and programming of the ECC Controller.

emac Address Map

Module Instance	Base Address	End Address
i_emac_emac0	0xFF800000	0xFF801FFF
i_emac_emac1	0xFF802000	0xFF803FFF
i_emac_emac2	0xFF804000	0xFF807FFF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_configuration on page 17-494	0x0	32	RW	0x0	Register 0 (MAC Configuration Register) The MAC Configuration register establishes receive and transmit operating modes.
gmacgrp_mac_frame_filter on page 17-508	0x4	32	RW	0x0	Register 1 (MAC Frame Filter) The MAC Frame Filter register contains the filter controls for receiving frames. Some of the controls from this register go to the address check block of the MAC, which performs the first level of address filtering. The second level of filtering is performed on the incoming frame, based on other controls such as Pass Bad Frames and Pass Control Frames.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_gmii_address on page 17-516	0x10	32	RW	0x0	Register 4 (GMII Address Register) The GMII Address register controls the management cycles to the external PHY through the management interface. Note: This register is present for all PHY interface when you select the Station Management (MDIO) feature in coreConsul-tant.
gmacgrp_gmii_data on page 17-520	0x14	32	RW	0x0	Register 5 (GMII Data Register) The GMII Data register stores Write data to be written to the PHY register located at the address specified in Register 4 (GMII Address Register). This register also stores the Read data from the PHY register located at the address specified by Register 4.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_flow_control on page 17-521	0x18	32	RW	0x0	Register 6 (Flow Control Register) The Flow Control register controls the generation and reception of the Control (Pause Command) frames by the MAC's Flow control module. A Write to a register with the Busy bit set to '1' triggers the Flow Control block to generate a Pause Control frame. The fields of the control frame are selected as specified in the 802.3x specification, and the Pause Time value from this register is used in the Pause Time field of the control frame. The Busy bit remains set until the control frame is transferred onto the cable. The Host must make sure that the Busy bit is cleared before writing to the register.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_vlan_tag on page 17-527	0x1C	32	RW	0x0	Register 7 (VLAN Tag Register) The VLAN Tag register contains the IEEE 802.1Q VLAN Tag to identify the VLAN frames. The MAC compares the 13th and 14th bytes of the receiving frame (Length/Type) with 16'h8100, and the following two bytes are compared with the VLAN tag. If a match occurs, the MAC sets the received VLAN bit in the receive frame status. The legal length of the frame is increased from 1,518 bytes to 1,522 bytes. If the VLAN Tag register is configured to be double-synchron-ized to the (G)MII clock domain, then consecutive writes to these register should be performed only after at least four clock cycles in the destination clock domain.
gmacgrp_version on page 17-531	0x20	32	RO	0x1037	Register 8 (Version Register) The Version registers identifies the version of the DWC_gmac.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_debug on page 17-531	0x24	32	RO	0x0	Register 9 (Debug Register) The Debug register gives the status of all main modules of the transmit and receive data-paths and the FIFOs. An all-zero status indicates that the MAC is in idle state (and FIFOs are empty) and no activity is going on in the data-paths. Note: The reset values, given for the Debug register, are valid only if the following clocks are present during the reset operation: * clk_csr_i, clk_app_i, hclk_i, or aclk_i * clk_tx_i * clk_rx_i
gmacgrp_lpi_control_status on page 17-537	0x30	32	RW	0x0	Register 12 (LPI Control and Status Register) The LPI Control and Status Register controls the LPI functions and provides the LPI interrupt status. The status bits are cleared when this register is read.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_lpi_timers_control on page 17-543	0x34	32	RW	0x3E80000	Register 13 (LPI Timers Control Register) The LPI Timers Control register controls the timeout values in the LPI states. It specifies the time for which the MAC transmits the LPI pattern and also the time for which the MAC waits before resuming the normal transmission.
gmacgrp_interrupt_status on page 17-544	0x38	32	RO	0x0	Register 14 (Interrupt Register) The Interrupt Status register identifies the events in the MAC that can generate interrupt. All interrupt events are generated only when the corresponding feature is enabled during operation. Therefore, these bits are reserved when the corresponding features are not present in the core.
gmacgrp_interrupt_mask on page 17-550	0x3C	32	RW	0x0	Register 15 (Interrupt Mask Register) The Interrupt Mask Register bits enable you to mask the interrupt signal because of the corresponding event in the Interrupt Status Register.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address0_high on page 17-552	0x40	32	RW	0x8000FFFF	Register 16 (MAC Address0 High Register) The MAC Address0 High register holds the upper 16 bits of the first 6-byte MAC address of the station. The first DA byte that is received on the (G)MII interface corresponds to the LS byte (Bits [7:0]) of the MAC Address Low register. For example, if 0x112233445566 is received (0x11 in lane 0 of the first column) on the (G)MII as the destination address, then the MacAddress0 Register [47:0] is compared with 0x665544332211. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address0 Low Register are written. For proper synchroniza-tion updates, the consecutive writes to this Address Low Register should be performed after at least four clock cycles in the destina-tion clock domain.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address0_low on page 17-553	0x44	32	RW	0xFFFFFFFF	Register 17 (MAC Address0 Low Register) The MAC Address0 Low register holds the lower 32 bits of the first 6-byte MAC address of the station.
gmacgrp_mac_address1_high on page 17-554	0x48	32	RW	0xFFFF	Register 18 (MAC Address1 High Register) The MAC Address1 High register holds the upper 16 bits of the second 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address1 Low Register are written. For proper synchroniza-tion updates, the consecutive writes to this Address Low Register should be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address1_low on page 17-559	0x4C	32	RW	0xFFFFFFFF	Register 19 (MAC Address1 Low Register) The MAC Address1 Low register holds the lower 32 bits of the second 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address2_high on page 17-560	0x50	32	RW	0xFFFF	Register 20 (MAC Address2 High Register) The MAC Address2 High register holds the upper 16 bits of the third 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address2 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address2 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address2_low on page 17-565	0x54	32	RW	0xFFFFFFFF	Register 21 (MAC Address2 Low Register) The MAC Address2 Low register holds the lower 32 bits of the third 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address3_high on page 17-565	0x58	32	RW	0xFFFF	Register 22 (MAC Address3 High Register) The MAC Address3 High register holds the upper 16 bits of the fourth 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address3 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address3 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address3_low on page 17-571	0x5C	32	RW	0xFFFFFFFF	Register 23 (MAC Address3 Low Register) The MAC Address3 Low register holds the lower 32 bits of the fourth 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address4_high on page 17-571	0x60	32	RW	0xFFFF	Register 24 (MAC Address4 High Register) The MAC Address4 High register holds the upper 16 bits of the fifth 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address4 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address4 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address4_low on page 17-577	0x64	32	RW	0xFFFFFFFF	Register 25 (MAC Address4 Low Register) The MAC Address4 Low register holds the lower 32 bits of the fifth 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address5_high on page 17-577	0x68	32	RW	0xFFFF	Register 26 (MAC Address5 High Register) The MAC Address5 High register holds the upper 16 bits of the sixth 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address5 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address5 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address5_low on page 17-583	0x6C	32	RW	0xFFFFFFFF	Register 27 (MAC Address5 Low Register) The MAC Address5 Low register holds the lower 32 bits of the sixth 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address6_high on page 17-583	0x70	32	RW	0xFFFF	Register 28 (MAC Address6 High Register) The MAC Address6 High register holds the upper 16 bits of the seventh 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address6 Low Register are written. For proper synchroniza-tion updates, the consecutive writes to this MAC Address6 Low Register should be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address6_low on page 17-588	0x74	32	RW	0xFFFFFFFF	Register 29 (MAC Address6 Low Register) The MAC Address6 Low register holds the lower 32 bits of the seventh 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address7_high on page 17-589	0x78	32	RW	0xFFFF	Register 30 (MAC Address7 High Register) The MAC Address7 High register holds the upper 16 bits of the eighth 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address7 Low Register are written. For proper synchroniza-tion updates, the consecutive writes to this MAC Address7 Low Register should be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address7_low on page 17-594	0x7C	32	RW	0xFFFFFFFF	Register 31 (MAC Address7 Low Register) The MAC Address7 Low register holds the lower 32 bits of the eighth 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address8_high on page 17-594	0x80	32	RW	0xFFFF	Register 32 (MAC Address8 High Register) The MAC Address8 High register holds the upper 16 bits of the nineth 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address8 Low Register are written. For proper synchroniza-tion updates, the consecutive writes to this MAC Address8 Low Register should be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address8_low on page 17-599	0x84	32	RW	0xFFFFFFFF	Register 33 (MAC Address8 Low Register) The MAC Address8 Low register holds the lower 32 bits of the nineth 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address9_high on page 17-600	0x88	32	RW	0xFFFF	Register 34 (MAC Address9 High Register) The MAC Address9 High register holds the upper 16 bits of the tenth 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address9 Low Register are written. For proper synchroniza-tion updates, the consecutive writes to this MAC Address9 Low Register should be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address9_low on page 17-605	0x8C	32	RW	0xFFFFFFFF	Register 35 (MAC Address9 Low Register) The MAC Address9 Low register holds the lower 32 bits of the tenth 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address10_high on page 17-605	0x90	32	RW	0xFFFF	Register 36 (MAC Address10 High Register) The MAC Address10 High register holds the upper 16 bits of the 11th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address10 Low Register are written. For proper synchroniza-tion updates, the consecutive writes to this MAC Address10 Low Register should be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address10_low on page 17-610	0x94	32	RW	0xFFFFFFFF	Register 37 (MAC Address10 Low Register) The MAC Address10 Low register holds the lower 32 bits of the 11th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address11_high on page 17-611	0x98	32	RW	0xFFFF	Register 38 (MAC Address11 High Register) The MAC Address11 High register holds the upper 16 bits of the 12th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address11 Low Register are written. For proper synchroniza-tion updates, the consecutive writes to this MAC Address11 Low Register should be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address11_low on page 17-616	0x9C	32	RW	0xFFFFFFFF	Register 39 (MAC Address1 Low Register) The MAC Address11 Low register holds the lower 32 bits of the 12th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address12_high on page 17-616	0xA0	32	RW	0xFFFF	Register 40 (MAC Address12 High Register) The MAC Address12 High register holds the upper 16 bits of the 13th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address13 Low Register are written. For proper synchroniza-tion updates, the consecutive writes to this MAC Address12 Low Register should be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address12_low on page 17-621	0xA4	32	RW	0xFFFFFFFF	Register 41 (MAC Address12 Low Register) The MAC Address12 Low register holds the lower 32 bits of the 13th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address13_high on page 17-622	0xA8	32	RW	0xFFFF	Register 42 (MAC Address13 High Register) The MAC Address13 High register holds the upper 16 bits of the 14th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address13 Low Register are written. For proper synchroniza-tion updates, the consecutive writes to this MAC Address13 Low Register should be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address13_low on page 17-627	0xAC	32	RW	0xFFFFFFFF	Register 43 (MAC Address13 Low Register) The MAC Address13 Low register holds the lower 32 bits of the 14th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address14_high on page 17-627	0xB0	32	RW	0xFFFF	Register 44 (MAC Address14 High Register) The MAC Address14 High register holds the upper 16 bits of the 15th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address15 Low Register are written. For proper synchroniza-tion updates, the consecutive writes to this MAC Address14 Low Register should be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address14_low on page 17-632	0xB4	32	RW	0xFFFFFFFF	Register 45 (MAC Address14 Low Register) The MAC Address14 Low register holds the lower 32 bits of the 15th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address15_high on page 17-633	0xB8	32	RW	0xFFFF	Register 46 (MAC Address15 High Register) The MAC Address15 High register holds the upper 16 bits of the 16th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address15 Low Register are written. For proper synchroniza-tion updates, the consecutive writes to this MAC Address15 Low Register should be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address15_low on page 17-637	0xBC	32	RW	0xFFFFFFFF	Register 47 (MAC Address15 Low Register) The MAC Address15 Low register holds the lower 32 bits of the 16th 6-byte MAC address of the station.
gmacgrp_sgmii_rgmii_smii_control_status on page 17-638	0xD8	32	RO	0x0	Register 54 (SGMII/ RGMII/SMII Status Register)
gmacgrp_wdog_timeout on page 17-640	0xDC	32	RW	0x0	Register 55 (Watchdog Timeout Register) This register controls the watchdog timeout for received frames.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_genpio on page 17-641	0xE0	32	RW	0x0	Register 56 (General Purpose IO Register) This register provides the control to drive up to 4 bits of output ports (GPO) and the status of up to 4 input ports (GPIS). It also provides the control to generate interrupts on events occurring on the gpi_i pin.
gmacgrp_mmc_control on page 17-644	0x100	32	RW	0x0	Register 64 (MMC Control Register) The MMC Control register establishes the operating mode of the management counters. Note: The bit 0 (Counters Reset) has higher priority than bit 4 (Counter Preset). Therefore, when the Software tries to set both bits in the same write cycle, all counters are cleared and the bit 4 is not set.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mmc_receive_interrupt on page 17-648	0x104	32	RO	0x0	Register 65 (MMC Receive Interrupt Register) The MMC Receive Interrupt register maintains the interrupts that are generated when the following happens: * Receive statistic counters reach half of their maximum values (0x8000_0000 for 32-bit counter and 0x8000 for 16-bit counter). * Receive statistic counters cross their maximum values (0xFFFF_FFFF for 32-bit counter and 0xFFFF for 16-bit counter). When the Counter Stop Rollover is set, then interrupts are set but the counter remains at all-ones. The MMC Receive Interrupt register is a 32-bit wide register. An interrupt bit is cleared when the respective MMC counter that caused the interrupt is read. The least significant byte lane (Bits[7:0]) of the respective counter must be read in order to clear the interrupt bit.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mmc_transmit_interrupt on page 17-656	0x108	32	RO	0x0	Register 66 (MMC Transmit Interrupt Register) The MMC Transmit Interrupt register maintains the interrupts generated when transmit statistic counters reach half of their maximum values (0x8000_0000 for 32-bit counter and 0x8000 for 16-bit counter), and the maximum values (0xFFFF_FFFF for 32-bit counter and 0xFFFF for 16-bit counter). When Counter Stop Rollover is set, then interrupts are set but the counter remains at all-ones. The MMC Transmit Interrupt register is a 32-bit wide register. An interrupt bit is cleared when the respective MMC counter that caused the interrupt is read. The least significant byte lane (Bits[7:0]) of the respective counter must be read in order to clear the interrupt bit.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mmc_receive_interrupt_mask on page 17-664	0x10C	32	RW	0x0	Regsiter 67 (MMC Receive Interrupt Mask Register) The MMC Receive Interrupt Mask register maintains the masks for the interrupts generated when the receive statistic counters reach half of their maximum value, or maximum value. This register is 32-bits wide.
gmacgrp_mmc_transmit_interrupt_mask on page 17-674	0x110	32	RW	0x0	Register 68 (MMC Transmit Interrupt Mask Register) The MMC Transmit Interrupt Mask register maintains the masks for the interrupts generated when the transmit statistic counters reach half of their maximum value or maximum value. This register is 32-bits wide.
gmacgrp_txoctet-count_gb on page 17-683	0x114	32	RO	0x0	Register 69 (Transmit Octet Count for Good and Bad Frames) This register maintains the number of bytes transmitted in good and bad frames exclusive of preamble and retried bytes.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_txframe-count_gb on page 17-684	0x118	32	RO	0x0	Register 70 (Transmit Frame Count for Good and Bad Frames) This register maintains the number of good and bad frames transmitted, exclusive of retried frames.
gmacgrp_txbroadcast-frames_g on page 17-685	0x11C	32	RO	0x0	Register 71 (Transmit Frame Count for Good Broadcast Frames) This register maintains the number of transmitted good broadcast frames.
gmacgrp_txmulticast-frames_g on page 17-686	0x120	32	RO	0x0	Register 72 (Transmit Frame Count for Good Multicast Frames) This register maintains the number of transmitted good multicast frames.
gmacgrp_tx64octets_gb on page 17-687	0x124	32	RO	0x0	Register 73 (Transmit Octet Count for Good and Bad 64 Byte Frames) This register maintains the number of transmitted good and bad frames with length of 64 bytes, exclusive of preamble and retried frames.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_tx65to127octets_gb on page 17-687	0x128	32	RO	0x0	Register 74 (Transmit Octet Count for Good and Bad 65 to 127 Bytes Frames) This register maintains the number of transmitted good and bad frames with length between 65 and 127 (inclusive) bytes, exclusive of preamble and retried frames.
gmacgrp_tx128to255octets_gb on page 17-688	0x12C	32	RO	0x0	Register 75 (Transmit Octet Count for Good and Bad 128 to 255 Bytes Frames) This register maintains the number of transmitted good and bad frames with length between 128 and 255 (inclusive) bytes, exclusive of preamble and retried frames.
gmacgrp_tx256to511octets_gb on page 17-689	0x130	32	RO	0x0	Register 76 (Transmit Octet Count for Good and Bad 256 to 511 Bytes Frames) This register maintains the number of transmitted good and bad frames with length between 256 and 511 (inclusive) bytes, exclusive of preamble and retried frames.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_tx512to1023octets_gb on page 17-690	0x134	32	RO	0x0	Register 77 (Transmit Octet Count for Good and Bad 512 to 1023 Bytes Frames) This register maintains the number of transmitted good and bad frames with length between 512 and 1,023 (inclusive) bytes, exclusive of preamble and retried frames.
gmacgrp_tx1024tomaxoctets_gb on page 17-691	0x138	32	RO	0x0	Register 78 (Transmit Octet Count for Good and Bad 1024 to Maxsize Bytes Frames) This register maintains the number of transmitted good and bad frames with length between 1,024 and maxsize (inclusive) bytes, exclusive of preamble and retried frames.
gmacgrp_txunicast-frames_gb on page 17-692	0x13C	32	RO	0x0	Register 79 (Transmit Frame Count for Good and Bad Unicast Frames) This register maintains the number of transmitted good and bad unicast frames.
gmacgrp_txmulticast-frames_gb on page 17-693	0x140	32	RO	0x0	Register 80 (Transmit Frame Count for Good and Bad Multicast Frames) This register maintains the number of transmitted good and bad multicast frames.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_txbroadcast-frames_gb on page 17-694	0x144	32	RO	0x0	Register 81 (Transmit Frame Count for Good and Bad Broadcast Frames) This register maintains the number of transmitted good and bad broadcast frames.
gmacgrp_txunderflo-werror on page 17-695	0x148	32	RO	0x0	Register 82 (Transmit Frame Count for Underflow Error Frames) This register maintains the number of frames aborted because of frame underflow error.
gmacgrp_txsinglecol_g on page 17-695	0x14C	32	RO	0x0	Register 83 (Transmit Frame Count for Frames Transmitted after Single Collision) This register maintains the number of successfully transmitted frames after a single collision in the half-duplex mode.
gmacgrp_txmulticol_g on page 17-696	0x150	32	RO	0x0	Register 84 (Transmit Frame Count for Frames Transmitted after Multiple Collision) This register maintains the number of successfully transmitted frames after multiple collisions in the half-duplex mode.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_txdeferred on page 17-697	0x154	32	RO	0x0	Register 85 (Transmit Frame Count for Deferred Frames) This register maintains the number of successfully transmitted frames after a deferral in the half-duplex mode.
gmacgrp_txlatecol on page 17-698	0x158	32	RO	0x0	Register 86 (Transmit Frame Count for Late Collision Error Frames) This register maintains the number of frames aborted because of late collision error.
gmacgrp_txexesscol on page 17-699	0x15C	32	RO	0x0	Register 87 (Transmit Frame Count for Excessive Collision Error Frames) This register maintains the number of frames aborted because of excessive (16) collision error.
gmacgrp_txcarriererr on page 17-699	0x160	32	RO	0x0	Register 88 (Transmit Frame Count for Carrier Sense Error Frames) This register maintains the number of frames aborted because of carrier sense error (no carrier or loss of carrier).

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_txoctetcnt on page 17-700	0x164	32	RO	0x0	Register 89 (Transmit Octet Count for Good Frames) This register maintains the number of bytes transmitted, exclusive of preamble, in good frames.
gmacgrp_txframe-count_g on page 17-701	0x168	32	RO	0x0	Register 90 (Transmit Frame Count for Good Frames) This register maintains the number of transmitted good frames, exclusive of preamble.
gmacgrp_txexcessdef on page 17-702	0x16C	32	RO	0x0	Register 91 (Transmit Frame Count for Excessive Deferral Error Frames) This register maintains the number of frames aborted because of excessive deferral error, that is, frames deferred for more than two max-sized frame times.
gmacgrp_txpauseframes on page 17-703	0x170	32	RO	0x0	Register 92 (Transmit Frame Count for Good PAUSE Frames) This register maintains the number of transmitted good PAUSE frames.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_txvlanframes_g on page 17-703	0x174	32	RO	0x0	Register 93 (Transmit Frame Count for Good VLAN Frames) This register maintains the number of transmitted good VLAN frames, exclusive of retried frames.
gmacgrp_txoversize_g on page 17-704	0x178	32	RO	0x0	Register 94 (Transmit Frame Count for Good Oversize Frames) This register maintains the number of transmitted good Oversize frames, exclusive of retried frames.
gmacgrp_rxframe-count_gb on page 17-705	0x180	32	RO	0x0	Register 96 (Receive Frame Count for Good and Bad Frames) This register maintains the number of received good and bad frames.
gmacgrp_rxoctet-count_gb on page 17-706	0x184	32	RO	0x0	Register 97 (Receive Octet Count for Good and Bad Frames) This register maintains the number of bytes received, exclusive of preamble, in good and bad frames.
gmacgrp_rxoctet-count_g on page 17-707	0x188	32	RO	0x0	Register 98 (Receive Octet Count for Good Frames) This register maintains the number of bytes received, exclusive of preamble, only in good frames.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_rxbroadcast-frames_g on page 17-708	0x18C	32	RO	0x0	Register 99 (Receive Frame Count for Good Broadcast Frames) This register maintains the number of received good broadcast frames.
gmacgrp_rxmulticast-frames_g on page 17-708	0x190	32	RO	0x0	Register 100 (Receive Frame Count for Good Multicast Frames) This register maintains the number of received good multicast frames.
gmacgrp_rxcrcerror on page 17-709	0x194	32	RO	0x0	Register 101 (Receive Frame Count for CRC Error Frames) This register maintains the number of frames received with CRC error.
gmacgrp_rxalignmen-terror on page 17-710	0x198	32	RO	0x0	Register 102 (Receive Frame Count for Alignment Error Frames) This register maintains the number of frames received with alignment (dribble) error. This field is valid only in the 10 or 100 Mbps mode.
gmacgrp_rxrunterror on page 17-711	0x19C	32	RO	0x0	Register 103 (Receive Frame Count for Runt Error Frames) This register maintains the number of frames received with runt error(<64 bytes and CRC error).

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_rxjabbererror on page 17-712	0x1A0	32	RO	0x0	Register 104 (Receive Frame Count for Jabber Error Frames) This register maintains the number of giant frames received with length (including CRC) greater than 1,518 bytes (1,522 bytes for VLAN tagged) and with CRC error. If Jumbo Frame mode is enabled, then frames of length greater than 9,018 bytes (9,022 for VLAN tagged) are considered as giant frames.
gmacgrp_rxundersize_g on page 17-713	0x1A4	32	RO	0x0	Register 105 (Receive Frame Count for Undersize Frames) This register maintains the number of frames received with length less than 64 bytes and without errors.
gmacgrp_rxoversize_g on page 17-713	0x1A8	32	RO	0x0	Register 106 (Receive Frame Count for Oversize Frames) This register maintains the number of frames received with length greater than the maxsize (1,518 or 1,522 for VLAN tagged frames) and without errors.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_rx64octets_gb on page 17-714	0x1AC	32	RO	0x0	Register 107 (Receive Frame Count for Good and Bad 64 Byte Frames) This register maintains the number of received good and bad frames with length 64 bytes, exclusive of preamble.
gmacgrp_rx65to127octets_gb on page 17-715	0x1B0	32	RO	0x0	Register 108 (Receive Frame Count for Good and Bad 65 to 127 Bytes Frames) This register maintains the number of received good and bad frames received with length between 65 and 127 (inclusive) bytes, exclusive of preamble.
gmacgrp_rx128to255octets_gb on page 17-716	0x1B4	32	RO	0x0	Register 109 (Receive Frame Count for Good and Bad 128 to 255 Bytes Frames) This register maintains the number of received good and bad frames with length between 128 and 255 (inclusive) bytes, exclusive of preamble.
gmacgrp_rx256to511octets_gb on page 17-717	0x1B8	32	RO	0x0	Register 110 (Receive Frame Count for Good and Bad 256 to 511 Bytes Frames) This register maintains the number of received good and bad frames with length between 256 and 511 (inclusive) bytes, exclusive of preamble.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_rx512to1023octets_gb on page 17-718	0x1BC	32	RO	0x0	Register 111 (Receive Frame Count for Good and Bad 512 to 1,023 Bytes Frames) This register maintains the number of received good and bad frames with length between 512 and 1,023 (inclusive) bytes, exclusive of preamble.
gmacgrp_rx1024tomaxoctets_gb on page 17-719	0x1C0	32	RO	0x0	Register 112 (Receive Frame Count for Good and Bad 1,024 to Maxsize Bytes Frames) This register maintains the number of received good and bad frames with length between 1,024 and maxsize (inclusive) bytes, exclusive of preamble.
gmacgrp_rxunicast-frames_g on page 17-720	0x1C4	32	RO	0x0	Register 113 (Receive Frame Count for Good Unicast Frames) This register maintains the number of received good unicast frames.
gmacgrp_rxlengtherror on page 17-721	0x1C8	32	RO	0x0	Register 114 (Receive Frame Count for Length Error Frames) This register maintains the number of frames received with length error (Length type field not equal to frame size) for all frames with valid length field.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_rxoutofrange-type on page 17-722	0x1CC	32	RO	0x0	Register 115 (Receive Frame Count for Out of Range Frames) This register maintains the number of received frames with length field not equal to the valid frame size (greater than 1,500 but less than 1,536).
gmacgrp_rxpauseframes on page 17-722	0x1D0	32	RO	0x0	Register 116 (Receive Frame Count for PAUSE Frames) This register maintains the number of received good and valid PAUSE frames.
gmacgrp_rxfifoover-flow on page 17-723	0x1D4	32	RO	0x0	Register 117 (Receive Frame Count for FIFO Overflow Frames) This register maintains the number of received frames missed because of FIFO overflow.
gmacgrp_rxvlanframes_gb on page 17-724	0x1D8	32	RO	0x0	Register 118 (Receive Frame Count for Good and Bad VLAN Frames) This register maintains the number of received good and bad VLAN frames.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_rxwatchdo-gerror on page 17-725	0x1DC	32	RO	0x0	Register 119 (Receive Frame Count for Watchdog Error Frames) This register maintains the number of frames received with error because of the watchdog timeout error (frames with more than 2,048 bytes or value programmed in Register 55 (Watchdog Timeout Register)).
gmacgrp_rxrcverror on page 17-726	0x1E0	32	RO	0x0	Register 120 (Receive Frame Count for Receive Error Frames) This register maintains the number of frames received with error because of the GMII/MII RXER error.
gmacgrp_rxctrlframes_g on page 17-726	0x1E4	32	RO	0x0	Register 121 (Receive Frame Count for Good Control Frames Frames) This register maintains the number of good control frames received.
gmacgrp_mmc_ipc_receive_interrupt_mask on page 17-727	0x200	32	RW	0x0	Register 128 (MMC Receive Checksum Offload Interrupt Mask Register)
gmacgrp_mmc_ipc_receive_interrupt on page 17-734	0x208	32	RO	0x0	Register 130 (MMC Receive Checksum Offload Interrupt Register)
gmacgrp_rxipv4_gd_frms on page 17-741	0x210	32	RO	0x0	Register 132 (rxipv4_ gd_frms Register)
gmacgrp_rxipv4_hdrerr_frms on page 17-741	0x214	32	RO	0x0	Register 133 (rxipv4_ hdrerr_frms Register)

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_rxipv4_nopay_frms on page 17-742	0x218	32	RO	0x0	Register 134 (rxipv4_nopay_frms Register)
gmacgrp_rxipv4_frag_frms on page 17-743	0x21C	32	RO	0x0	Register 135 (rxipv4_frag_frms Register)
gmacgrp_rxipv4_udsbl_frms on page 17-743	0x220	32	RO	0x0	Register 136 (rxipv4_udsbl_frms Register)
gmacgrp_rxipv6_gd_frms on page 17-744	0x224	32	RO	0x0	Register 137 (rxipv6_gd_frms Register)
gmacgrp_rxipv6_hdrerr_frms on page 17-745	0x228	32	RO	0x0	Register 138 (rxipv6_hdrerr_frms Register)
gmacgrp_rxipv6_nopay_frms on page 17-746	0x22C	32	RO	0x0	Register 139 (rxipv6_nopay_frms)
gmacgrp_rxudp_gd_frms on page 17-747	0x230	32	RO	0x0	Register 140 (rxudp_gd_frms Register)
gmacgrp_rxudp_err_frms on page 17-747	0x234	32	RO	0x0	Register 141 (rxudp_err_frms Register)
gmacgrp_rxtcp_gd_frms on page 17-748	0x238	32	RO	0x0	Register 142 (rxtcp_gd_frms Register)
gmacgrp_rxtcp_err_frms on page 17-749	0x23C	32	RO	0x0	Register 143 (rxtcp_err_frms Register)
gmacgrp_rxicmp_gd_frms on page 17-750	0x240	32	RO	0x0	Register 144 (rxicmp_gd_frms Register)
gmacgrp_rxicmp_err_frms on page 17-750	0x244	32	RO	0x0	Register 145 (rxicmp_err_frms Register)
gmacgrp_rxipv4_gd_octets on page 17-751	0x250	32	RO	0x0	Register 148 (rxipv4_gd_octets Register)
gmacgrp_rxipv4_hdrerr_octets on page 17-752	0x254	32	RO	0x0	Register 149 (rxipv4_hdrerr_octets)
gmacgrp_rxipv4_nopay_octets on page 17-753	0x258	32	RO	0x0	Register 150 (rxipv4_nopay_octets Register)
gmacgrp_rxipv4_frag_octets on page 17-754	0x25C	32	RO	0x0	Register 151 (rxipv4_frag_octets Register)

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_rxipv4_udsbl_octets on page 17-754	0x260	32	RO	0x0	Register 152 (rxipv4_udsbl_octets Register)
gmacgrp_rxipv6_gd_octets on page 17-755	0x264	32	RO	0x0	Register 153 (rxipv6_gd_octets Register)
gmacgrp_rxipv6_hdrerr_octets on page 17-756	0x268	32	RO	0x0	Register 154 (rxipv6_hdrerr_octets Register)
gmacgrp_rxipv6_nopay_octets on page 17-757	0x26C	32	RO	0x0	Register 155 (rxipv6_nopay_octets Register)
gmacgrp_rxudp_gd_octets on page 17-758	0x270	32	RO	0x0	Register 156 (rxudp_gd_octets Register)
gmacgrp_rxudp_err_octets on page 17-758	0x274	32	RO	0x0	Register 157 (rxudp_err_octets Register)
gmacgrp_rxtcp_gd_octets on page 17-759	0x278	32	RO	0x0	Register 158 (rxtcp_gd_octets Register)
gmacgrp_rxtcperroc-tets on page 17-760	0x27C	32	RO	0x0	Register 159 (rxtcp_err_octets Register)
gmacgrp_rxicmp_gd_octets on page 17-760	0x280	32	RO	0x0	Register 160 (rxicmp_gd_octets Register)
gmacgrp_rxicmp_err_octets on page 17-761	0x284	32	RO	0x0	Register 161 (rxicmp_err_octets Register)
gmacgrp_l3_l4_control0 on page 17-762	0x400	32	RW	0x0	Register 256 (Layer 3 and Layer 4 Control Register 0) This register controls the operations of the filter 0 of Layer 3 and Layer 4.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_layer4_address0 on page 17-768	0x404	32	RW	0x0	Register 257 (Layer 4 Address Register 0) You can configure the Layer 3 and Layer 4 Address Registers to be double-synchronized by selecting the Synchronize Layer 3 and Layer 4 Address Registers to Rx Clock Domain option in coreConsultant. If the Layer 3 and Layer 4 Address Registers are configured to be double-synchronized to the Rx clock domains, then the synchroniza-tion is triggered only when Bits[31:24] (in little-endian mode) of the Layer 3 and Layer 4 Address Registers are written. For proper synchronization updates, you should perform the consecu-tive writes to the same Layer 3 and Layer 4 Address Registers after at least four clock cycles delay of the destination clock.
gmacgrp_layer3_addr0_reg0 on page 17-770	0x410	32	RW	0x0	Register 260 (Layer 3 Address 0 Register 0) For IPv4 frames, the Layer 3 Address 0 Register 0 contains the 32-bit IP Source Address field. For IPv6 frames, it contains Bits[31:0] of the 128-bit IP Source Address or Destination Address field.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_layer3_addr1_reg0 on page 17-771	0x414	32	RW	0x0	Register 261 (Layer 3 Address 1 Register 0) For IPv4 frames, the Layer 3 Address 1 Register 0 contains the 32-bit IP Destination Address field. For IPv6 frames, it contains Bits[63:32] of the 128-bit IP Source Address or Destination Address field.
gmacgrp_layer3_addr2_reg0 on page 17-772	0x418	32	RW	0x0	Register 262 (Layer 3 Address 2 Register 0) For IPv4 frames, the Layer 3 Address 2 Register 0 is reserved. For IPv6 frames, it contains Bits [95:64] of the 128-bit IP Source Address or Destination Address field.
gmacgrp_layer3_addr3_reg0 on page 17-774	0x41C	32	RW	0x0	Register 263 (Layer 3 Address 3 Register 0) For IPv4 frames, the Layer 3 Address 3 Register 0 is reserved. For IPv6 frames, it contains Bits [127:96] of the 128-bit IP Source Address or Destination Address field.
gmacgrp_l3_l4_control1 on page 17-775	0x430	32	RW	0x0	Register 268 (Layer 3 and Layer 4 Control Register 1)
gmacgrp_layer4_address1 on page 17-779	0x434	32	RW	0x0	Register 269 (Layer 4 Address Register 1)
gmacgrp_layer3_addr0_reg1 on page 17-781	0x440	32	RW	0x0	Register 272 (Layer 3 Address 0 Register 1)

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_layer3_addr1_reg1 on page 17-782	0x444	32	RW	0x0	Register 273 (Layer 3 Address 1 Register 1)
gmacgrp_layer3_addr2_reg1 on page 17-783	0x448	32	RW	0x0	Register 274 (Layer 3 Address 2 Register 1)
gmacgrp_layer3_addr3_reg1 on page 17-784	0x44C	32	RW	0x0	Register 275 (Layer 3 Address 3 Register 1)
gmacgrp_l3_l4_control2 on page 17-785	0x460	32	RW	0x0	Register 280 (Layer 3 and Layer 4 Control Register 2)
gmacgrp_layer4_address2 on page 17-790	0x464	32	RW	0x0	Register 281 (Layer 4 Address Register 2)
gmacgrp_layer3_addr0_reg2 on page 17-792	0x470	32	RW	0x0	Register 284 (Layer 3 Address 0 Register 2)
gmacgrp_layer3_addr1_reg2 on page 17-793	0x474	32	RW	0x0	Register 285 (Layer 3 Address 1 Register 2)
gmacgrp_layer3_addr2_reg2 on page 17-794	0x478	32	RW	0x0	Register 286 (Layer 3 Address 2 Register 2)
gmacgrp_layer3_addr3_reg2 on page 17-795	0x47C	32	RW	0x0	Register 287 (Layer 3 Address 3 Register 2)
gmacgrp_l3_l4_control3 on page 17-797	0x490	32	RW	0x0	Register 292 (Layer 3 and Layer 4 Control Register 3)
gmacgrp_layer4_address3 on page 17-801	0x494	32	RW	0x0	Register 293 (Layer 4 Address Register 3)
gmacgrp_layer3_addr0_reg3 on page 17-803	0x4A0	32	RW	0x0	Register 296 (Layer 3 Address 0 Register 3)
gmacgrp_layer3_addr1_reg3 on page 17-804	0x4A4	32	RW	0x0	Register 297 (Layer 3 Address 1 Register 3)
gmacgrp_layer3_addr2_reg3 on page 17-805	0x4A8	32	RW	0x0	Register 298 (Layer 3 Address 2 Register 3)
gmacgrp_layer3_addr3_reg3 on page 17-806	0x4AC	32	RW	0x0	Register 299 (Layer 3 Address 3 Register 3)
gmacgrp_hash_table_reg0 on page 17-807	0x500	32	RW	0x0	Register 320 (Hash Table Register 0)

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_hash_table_reg1 on page 17-809	0x504	32	RW	0x0	Register 321 (Hash Table Register 1)
gmacgrp_hash_table_reg2 on page 17-809	0x508	32	RW	0x0	Register 322 (Hash Table Register 2)
gmacgrp_hash_table_reg3 on page 17-810	0x50C	32	RW	0x0	Register 323 (Hash Table Register 3)
gmacgrp_hash_table_reg4 on page 17-811	0x510	32	RW	0x0	Register 324 (Hash Table Register 4)
gmacgrp_hash_table_reg5 on page 17-812	0x514	32	RW	0x0	Register 325 (Hash Table Register 5)
gmacgrp_hash_table_reg6 on page 17-812	0x518	32	RW	0x0	Register 326 (Hash Table Register 6)
gmacgrp_hash_table_reg7 on page 17-813	0x51C	32	RW	0x0	Register 327 (Hash Table Register 7)
gmacgrp_vlan_incl_reg on page 17-814	0x584	32	RW	0x0	Register 353 (VLAN Tag Inclusion or Replacement Register) The VLAN Tag Inclusion or Replacement register contains the VLAN tag for insertion or replacement in the transmit frames.
gmacgrp_vlan_hash_table_reg on page 17-816	0x588	32	RW	0x0	Register 354 (VLAN Hash Table Register)

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_timestamp_control on page 17-818	0x700	32	RW	0x2000	Register 448 (Timestamp Control Register) This register controls the operation of the System Time generator and the processing of PTP packets for timestamping in the Receiver. Note: * Bits[5:1] are reserved when External Timestamp Input feature is enabled. * Bits[19:8] are reserved and read-only when Advanced Timestamp feature is not enabled. * Bits[28:24] are reserved and read-only when Auxiliary Snapshot feature is not enabled. * Release 3.60a onwards, the functions of Bits 17 and 16 (SNAPTYPSEL) have changed. These functions are not backward compatible with the functions described in release 3.50a.
gmacgrp_sub_second_increment on page 17-826	0x704	32	RW	0x0	Register 449 (Sub-Second Increment Register)
gmacgrp_system_time_seconds on page 17-827	0x708	32	RO	0x0	Register 450 (System Time - Seconds Register)
gmacgrp_system_time_nanoseconds on page 17-828	0x70C	32	RO	0x0	Register 451 (System Time - Nanoseconds Register)

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_system_time_seconds_update on page 17-829	0x710	32	RW	0x0	Register 452 (System Time - Seconds Update Register)
gmacgrp_system_time_nanoseconds_update on page 17-830	0x714	32	RW	0x0	Register 453 (System Time - Nanoseconds Update Register)
gmacgrp_timestamp_addend on page 17-831	0x718	32	RW	0x0	Register 454 (Timestamp Addend Register)
gmacgrp_target_time_seconds on page 17-832	0x71C	32	RW	0x0	Register 455 (Target Time Seconds Register)
gmacgrp_target_time_nanoseconds on page 17-833	0x720	32	RW	0x0	Register 456 (Target Time Nanoseconds Register)
gmacgrp_system_time_higher_word_seconds on page 17-835	0x724	32	RW	0x0	Register 457 (System Time - Higher Word Seconds Register)
gmacgrp_timestamp_status on page 17-836	0x728	32	RO	0x0	Register 458 (Timestamp Status Register)
gmacgrp_pps_control on page 17-838	0x72C	32	RW	0x0	Register 459 (PPS Control Register)
gmacgrp_auxiliary_timestamp_nanoseconds on page 17-842	0x730	32	RO	0x0	Register 460 (Auxiliary Timestamp - Nanoseconds Register)
gmacgrp_auxiliary_timestamp_seconds on page 17-842	0x734	32	RO	0x0	Register 461 (Auxiliary Timestamp - Seconds Register)
gmacgrp_pps0_interval on page 17-843	0x760	32	RW	0x0	Register 472 (PPS0 Interval Register)
gmacgrp_pps0_width on page 17-844	0x764	32	RW	0x0	Register 473 (PPS0 Width Register)

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address16_high on page 17-845	0x800	32	RW	0xFFFF	Register 512 (MAC Address16 High Register) The MAC Address16 High register holds the upper 16 bits of the 17th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address16 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address16 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address16_low on page 17-850	0x804	32	RW	0xFFFFFFFF	Register 513 (MAC Address16 Low Register) The MAC Address16 Low register holds the lower 32 bits of the 17th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address17_high on page 17-851	0x808	32	RW	0xFFFF	Register 514 (MAC Address17 High Register) The MAC Address17 High register holds the upper 16 bits of the 18th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address17 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address17 Low Register must be performed after at least four clock cycles in the destination clock domain.
gmacgrp_mac_address17_low on page 17-856	0x80C	32	RW	0xFFFFFFFF	Register 515 (MAC Address17 Low Register) The MAC Address17 Low register holds the lower 32 bits of the 18th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address18_high on page 17-856	0x810	32	RW	0xFFFF	Register 516 (MAC Address18 High Register) The MAC Address18 High register holds the upper 16 bits of the 19th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address18 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address18 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address18_low on page 17-862	0x814	32	RW	0xFFFFFFFF	Register 517 (MAC Address18 Low Register) The MAC Address18 Low register holds the lower 32 bits of the 19th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address19_high on page 17-862	0x818	32	RW	0xFFFF	Register 518 (MAC Address19 High Register) The MAC Address19 High register holds the upper 16 bits of the 20th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address19 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address19 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address19_low on page 17-868	0x81C	32	RW	0xFFFFFFFF	Register 519 (MAC Address19 Low Register) The MAC Address19 Low register holds the lower 32 bits of the 20th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address20_high on page 17-868	0x820	32	RW	0xFFFF	Register 520 (MAC Address20 High Register) The MAC Address20 High register holds the upper 16 bits of the 21st 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address20 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address20 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address20_low on page 17-874	0x824	32	RW	0xFFFFFFFF	Register 521 (MAC Address20 Low Register) The MAC Address20 Low register holds the lower 32 bits of the 21st 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address21_high on page 17-874	0x828	32	RW	0xFFFF	Register 522 (MAC Address21 High Register) The MAC Address21 High register holds the upper 16 bits of the 22nd 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address21 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address21 Low Register must be performed after at least four clock cycles in the destination clock domain.
gmacgrp_mac_address21_low on page 17-880	0x82C	32	RW	0xFFFFFFFF	Register 523 (MAC Address21 Low Register) The MAC Address21 Low register holds the lower 32 bits of the 22nd 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address22_high on page 17-880	0x830	32	RW	0xFFFF	Register 524 (MAC Address22 High Register) The MAC Address22 High register holds the upper 16 bits of the 23rd 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address22 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address22 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address22_low on page 17-886	0x834	32	RW	0xFFFFFFFF	Register 525 (MAC Address22 Low Register) The MAC Address22 Low register holds the lower 32 bits of the 23rd 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address23_high on page 17-886	0x838	32	RW	0xFFFF	Register 526 (MAC Address23 High Register The MAC Address23 High register holds the upper 16 bits of the 24th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address23 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address23 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address23_low on page 17-892	0x83C	32	RW	0xFFFFFFFF	Register 527 (MAC Address23 Low Register) The MAC Address23 Low register holds the lower 32 bits of the 24th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address24_high on page 17-892	0x840	32	RW	0xFFFF	Register 528 (MAC Address24 High Register) The MAC Address24 High register holds the upper 16 bits of the 25th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address24 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address24 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address24_low on page 17-898	0x844	32	RW	0xFFFFFFFF	Register 529 (MAC Address24 Low Register) The MAC Address24 Low register holds the lower 32 bits of the 25th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address25_high on page 17-898	0x848	32	RW	0xFFFF	Register 530 (MAC Address25 High Register) The MAC Address25 High register holds the upper 16 bits of the 6-byte 26th MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address25 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address25 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address25_low on page 17-904	0x84C	32	RW	0xFFFFFFFF	Register 531 (MAC Address25 Low Register) The MAC Address25 Low register holds the lower 32 bits of the 26th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address26_high on page 17-904	0x850	32	RW	0xFFFF	Register 532 (MAC Address26 High Register) The MAC Address26 High register holds the upper 16 bits of the 27th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address26 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address26 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address26_low on page 17-910	0x854	32	RW	0xFFFFFFFF	Register 533 (MAC Address26 Low Register) The MAC Address26 Low register holds the lower 32 bits of the 27th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address27_high on page 17-910	0x858	32	RW	0xFFFF	Register 534 (MAC Address27 High Register) The MAC Address27 High register holds the upper 16 bits of the 28th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address27 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address27 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address27_low on page 17-916	0x85C	32	RW	0xFFFFFFFF	Register 535 (MAC Address27 Low Register) The MAC Address27 Low register holds the lower 32 bits of the 28th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address28_high on page 17-916	0x860	32	RW	0xFFFF	Register 536 (MAC Address28 High Register) The MAC Address28 High register holds the upper 16 bits of the 29th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address28 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address28 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address28_low on page 17-922	0x864	32	RW	0xFFFFFFFF	Register 537 (MAC Address28 Low Register) The MAC Address28 Low register holds the lower 32 bits of the 29th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address29_high on page 17-922	0x868	32	RW	0xFFFF	Register 538 (MAC Address29 High Register) The MAC Address29 High register holds the upper 16 bits of the 6-byte 30th MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address29 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address29 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address29_low on page 17-928	0x86C	32	RW	0xFFFFFFFF	Register 539 (MAC Address29 Low Register) The MAC Address29 Low register holds the lower 32 bits of the 30th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address30_high on page 17-928	0x870	32	RW	0xFFFF	Register 540 (MAC Address30 High Register) The MAC Address30 High register holds the upper 16 bits of the 31st 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address30 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address30 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address30_low on page 17-934	0x874	32	RW	0xFFFFFFFF	Register 541 (MAC Address30 Low Register) The MAC Address30 Low register holds the lower 32 bits of the 31st 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address31_high on page 17-934	0x878	32	RW	0xFFFF	Register 542 (MAC Address31 High Register) The MAC Address31 High register holds the upper 16 bits of the 32nd 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address31 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address31 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address31_low on page 17-939	0x87C	32	RW	0xFFFFFFFF	Register 543 (MAC Address31 Low Register) The MAC Address31 Low register holds the lower 32 bits of the 32nd 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address32_high on page 17-940	0x880	32	RW	0xFFFF	Register 544 (MAC Address32 High Register) The MAC Address32 High register holds the upper 16 bits of the 33rd 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address32 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address32 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address32_low on page 17-941	0x884	32	RW	0xFFFFFFFF	Register 545 (MAC Address32 Low Register) The MAC Address32 Low register holds the lower 32 bits of the 33rd 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address33_high on page 17-942	0x888	32	RW	0xFFFF	Register 546 (MAC Address33 High Register) The MAC Address33 High register holds the upper 16 bits of the 34th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address33 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address33 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address33_low on page 17-944	0x88C	32	RW	0xFFFFFFFF	Register 547 (MAC Address33 Low Register) The MAC Address33 Low register holds the lower 32 bits of the 34th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address34_high on page 17-944	0x890	32	RW	0xFFFF	Register 548 (MAC Address34 High Register) The MAC Address34 High register holds the upper 16 bits of the 35th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address34 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address34 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address34_low on page 17-946	0x894	32	RW	0xFFFFFFFF	Register 549 (MAC Address34 Low Register) The MAC Address34 Low register holds the lower 32 bits of the 35th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address35_high on page 17-947	0x898	32	RW	0xFFFF	Register 550 (MAC Address35 High Register) The MAC Address35 High register holds the upper 16 bits of the 36th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address35 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address35 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address35_low on page 17-948	0x89C	32	RW	0xFFFFFFFF	Register 551 (MAC Address35 Low Register) The MAC Address35 Low register holds the lower 32 bits of the 36th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address36_high on page 17-949	0x8A0	32	RW	0xFFFF	Register 552 (MAC Address36 High Register) The MAC Address36 High register holds the upper 16 bits of the 37th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address36 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address36 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address36_low on page 17-951	0x8A4	32	RW	0xFFFFFFFF	Register 553 (MAC Address36 Low Register) The MAC Address36 Low register holds the lower 32 bits of the 34th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address37_high on page 17-951	0x8A8	32	RW	0xFFFF	Register 554 (MAC Address37 High Register) The MAC Address37 High register holds the upper 16 bits of the 38th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address37 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address37 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address37_low on page 17-953	0x8AC	32	RW	0xFFFFFFFF	Register 555 (MAC Address37 Low Register) The MAC Address37 Low register holds the lower 32 bits of the 37th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address38_high on page 17-954	0x8B0	32	RW	0xFFFF	Register 556 (MAC Address38 High Register) The MAC Address38 High register holds the upper 16 bits of the 39th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address38 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address38 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address38_low on page 17-955	0x8B4	32	RW	0xFFFFFFFF	Register 557 (MAC Address38 Low Register) The MAC Address38 Low register holds the lower 32 bits of the 39th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address39_high on page 17-956	0x8B8	32	RW	0xFFFF	Register 558 (MAC Address39 High Register) The MAC Address39 High register holds the upper 16 bits of the 40th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address40 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address40 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address39_low on page 17-958	0x8BC	32	RW	0xFFFFFFFF	Register 559 (MAC Address39 Low Register) The MAC Address39 Low register holds the lower 32 bits of the 40th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address40_high on page 17-958	0x8C0	32	RW	0xFFFF	Register 560 (MAC Address40 High Register) The MAC Address40 High register holds the upper 16 bits of the 41st 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address40 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address40 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address40_low on page 17-960	0x8C4	32	RW	0xFFFFFFFF	Register 561 (MAC Address40 Low Register) The MAC Address40 Low register holds the lower 32 bits of the 41st 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address41_high on page 17-961	0x8C8	32	RW	0xFFFF	Register 562 (MAC Address41 High Register) The MAC Address41 High register holds the upper 16 bits of the 42nd 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address41 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address41 Low Register must be performed after at least four clock cycles in the destination clock domain.
gmacgrp_mac_address41_low on page 17-962	0x8CC	32	RW	0xFFFFFFFF	Register 563 (MAC Address41 Low Register) The MAC Address41 Low register holds the lower 32 bits of the 42nd 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address42_high on page 17-963	0x8D0	32	RW	0xFFFF	Register 564 (MAC Address42 High Register) The MAC Address42 High register holds the upper 16 bits of the 43rd 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address42 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address42 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address42_low on page 17-965	0x8D4	32	RW	0xFFFFFFFF	Register 565 (MAC Address42 Low Register) The MAC Address42 Low register holds the lower 32 bits of the 43rd 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address43_high on page 17-965	0x8D8	32	RW	0xFFFF	Register 566 (MAC Address43 High Register) The MAC Address43 High register holds the upper 16 bits of the 44th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address43 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address43 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address43_low on page 17-967	0x8DC	32	RW	0xFFFFFFFF	Register 567 (MAC Address43 Low Register) The MAC Address43 Low register holds the lower 32 bits of the 44th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address44_high on page 17-968	0x8E0	32	RW	0xFFFF	Register 568 (MAC Address44 High Register) The MAC Address44 High register holds the upper 16 bits of the 45th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address44 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address44 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address44_low on page 17-969	0x8E4	32	RW	0xFFFFFFFF	Register 569 (MAC Address44 Low Register) The MAC Address44 Low register holds the lower 32 bits of the 45th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address45_high on page 17-970	0x8E8	32	RW	0xFFFF	Register 570 (MAC Address45 High Register) The MAC Address45 High register holds the upper 16 bits of the 46th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address45 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address45 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address45_low on page 17-972	0x8EC	32	RW	0xFFFFFFFF	Register 571 (MAC Address45 Low Register) The MAC Address45 Low register holds the lower 32 bits of the 46th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address46_high on page 17-972	0x8F0	32	RW	0xFFFF	Register 572 (MAC Address46 High Register) The MAC Address46 High register holds the upper 16 bits of the 47th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address46 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address46 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address46_low on page 17-974	0x8F4	32	RW	0xFFFFFFFF	Register 573 (MAC Address46 Low Register) The MAC Address46 Low register holds the lower 32 bits of the 47th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address47_high on page 17-975	0x8F8	32	RW	0xFFFF	Register 574 (MAC Address47 High Register) The MAC Address47 High register holds the upper 16 bits of the 48th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address47 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address47 Low Register must be performed after at least four clock cycles in the destination clock domain.
gmacgrp_mac_address47_low on page 17-976	0x8FC	32	RW	0xFFFFFFFF	Register 575 (MAC Address47 Low Register) The MAC Address47 Low register holds the lower 32 bits of the 48th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address48_high on page 17-977	0x900	32	RW	0xFFFF	Register 576 (MAC Address48 High Register) The MAC Address48 High register holds the upper 16 bits of the 49th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address48 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address48 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address48_low on page 17-979	0x904	32	RW	0xFFFFFFFF	Register 577 (MAC Address48 Low Register) The MAC Address48 Low register holds the lower 32 bits of the 49th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address49_high on page 17-979	0x908	32	RW	0xFFFF	Register 578 (MAC Address49 High Register) The MAC Address49 High register holds the upper 16 bits of the 50th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address49 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address49 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address49_low on page 17-981	0x90C	32	RW	0xFFFFFFFF	Register 579 (MAC Address49 Low Register) The MAC Address49 Low register holds the lower 32 bits of the 50th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address50_high on page 17-982	0x910	32	RW	0xFFFF	Register 580 (MAC Address50 High Register) The MAC Address50 High register holds the upper 16 bits of the 51st 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address50 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address50 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address50_low on page 17-983	0x914	32	RW	0xFFFFFFFF	Register 581 (MAC Address50 Low Register) The MAC Address50 Low register holds the lower 32 bits of the 51st 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address51_high on page 17-984	0x918	32	RW	0xFFFF	Register 582 (MAC Address51 High Register) The MAC Address51 High register holds the upper 16 bits of the 52nd 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address51 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address51 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address51_low on page 17-986	0x91C	32	RW	0xFFFFFFFF	Register 583 (MAC Address51 Low Register) The MAC Address51 Low register holds the lower 32 bits of the 52nd 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address52_high on page 17-986	0x920	32	RW	0xFFFF	Register 584 (MAC Address52 High Register) The MAC Address52 High register holds the upper 16 bits of the 53rd 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address52 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address52 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address52_low on page 17-988	0x924	32	RW	0xFFFFFFFF	Register 585 (MAC Address52 Low Register) The MAC Address52 Low register holds the lower 32 bits of the 53rd 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address53_high on page 17-989	0x928	32	RW	0xFFFF	Register 586 (MAC Address53 High Register) The MAC Address53 High register holds the upper 16 bits of the 54th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address53 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address53 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address53_low on page 17-990	0x92C	32	RW	0xFFFFFFFF	Register 587 (MAC Address53 Low Register) The MAC Address53 Low register holds the lower 32 bits of the 54th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address54_high on page 17-991	0x930	32	RW	0xFFFF	Register 588 (MAC Address54 High Register) The MAC Address54 High register holds the upper 16 bits of the 55th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address54 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address54 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address54_low on page 17-993	0x934	32	RW	0xFFFFFFFF	Register 589 (MAC Address54 Low Register) The MAC Address54 Low register holds the lower 32 bits of the 55th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address55_high on page 17-993	0x938	32	RW	0xFFFF	Register 590 (MAC Address55 High Register) The MAC Address55 High register holds the upper 16 bits of the 56th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address55 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address55 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address55_low on page 17-995	0x93C	32	RW	0xFFFFFFFF	Register 591 (MAC Address55 Low Register) The MAC Address55 Low register holds the lower 32 bits of the 56th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address56_high on page 17-996	0x940	32	RW	0xFFFF	Register 592 (MAC Address56 High Register) The MAC Address56 High register holds the upper 16 bits of the 57th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address56 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address56 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address56_low on page 17-997	0x944	32	RW	0xFFFFFFFF	Register 593 (MAC Address56 Low Register) The MAC Address56 Low register holds the lower 32 bits of the 57th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address57_high on page 17-998	0x948	32	RW	0xFFFF	Register 594 (MAC Address57 High Register) The MAC Address57 High register holds the upper 16 bits of the 58th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address57 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address57 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address57_low on page 17-1000	0x94C	32	RW	0xFFFFFFFF	Register 595 (MAC Address57 Low Register) The MAC Address57 Low register holds the lower 32 bits of the 58th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address58_high on page 17-1000	0x950	32	RW	0xFFFF	Register 596 (MAC Address58 High Register) The MAC Address58 High register holds the upper 16 bits of the 59th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address58 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address58 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address58_low on page 17-1002	0x954	32	RW	0xFFFFFFFF	Register 597 (MAC Address58 Low Register) The MAC Address58 Low register holds the lower 32 bits of the 59th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address59_high on page 17-1003	0x958	32	RW	0xFFFF	Register 598 (MAC Address59 High Register) The MAC Address59 High register holds the upper 16 bits of the 60th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address59 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address59 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address59_low on page 17-1004	0x95C	32	RW	0xFFFFFFFF	Register 599 (MAC Address59 Low Register) The MAC Address59 Low register holds the lower 32 bits of the 60th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address60_high on page 17-1005	0x960	32	RW	0xFFFF	Register 600 (MAC Address60 High Register) The MAC Address60 High register holds the upper 16 bits of the 61st 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address60 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address60 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address60_low on page 17-1007	0x964	32	RW	0xFFFFFFFF	Register 601 (MAC Address60 Low Register) The MAC Address60 Low register holds the lower 32 bits of the 61st 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address61_high on page 17-1007	0x968	32	RW	0xFFFF	Register 602 (MAC Address61 High Register) The MAC Address61 High register holds the upper 16 bits of the 62nd 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address61 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address61 Low Register must be performed after at least four clock cycles in the destination clock domain.
gmacgrp_mac_address61_low on page 17-1009	0x96C	32	RW	0xFFFFFFFF	Register 603 (MAC Address61 Low Register) The MAC Address61 Low register holds the lower 32 bits of the 62nd 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address62_high on page 17-1010	0x970	32	RW	0xFFFF	Register 604 (MAC Address62 High Register) The MAC Address62 High register holds the upper 16 bits of the 63rd 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address62 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address62 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address62_low on page 17-1011	0x974	32	RW	0xFFFFFFFF	Register 605 (MAC Address62 Low Register) The MAC Address62 Low register holds the lower 32 bits of the 63rd 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address63_high on page 17-1012	0x978	32	RW	0xFFFF	Register 606 (MAC Address63 High Register) The MAC Address63 High register holds the upper 16 bits of the 64th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address63 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address63 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address63_low on page 17-1014	0x97C	32	RW	0xFFFFFFFF	Register 607 (MAC Address63 Low Register) The MAC Address63 Low register holds the lower 32 bits of the 64th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address64_high on page 17-1014	0x980	32	RW	0xFFFF	Register 608 (MAC Address64 High Register) The MAC Address64 High register holds the upper 16 bits of the 65th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address64 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address64 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address64_low on page 17-1016	0x984	32	RW	0xFFFFFFFF	Register 609 (MAC Address64 Low Register) The MAC Address64 Low register holds the lower 32 bits of the 65th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address65_high on page 17-1017	0x988	32	RW	0xFFFF	Register 610 (MAC Address65 High Register) The MAC Address65 High register holds the upper 16 bits of the 66th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address65 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address65 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address65_low on page 17-1018	0x98C	32	RW	0xFFFFFFFF	Register 611 (MAC Address65 Low Register) The MAC Address65 Low register holds the lower 32 bits of the 66th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address66_high on page 17-1019	0x990	32	RW	0xFFFF	Register 612 (MAC Address66 High Register) The MAC Address66 High register holds the upper 16 bits of the 67th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address66 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address66 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address66_low on page 17-1021	0x994	32	RW	0xFFFFFFFF	Register 613 (MAC Address66 Low Register) The MAC Address66 Low register holds the lower 32 bits of the 67th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address67_high on page 17-1021	0x998	32	RW	0xFFFF	Register 614 (MAC Address67 High Register) The MAC Address67 High register holds the upper 16 bits of the 68th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address67 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address67 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address67_low on page 17-1023	0x99C	32	RW	0xFFFFFFFF	Register 615 (MAC Address67 Low Register) The MAC Address67 Low register holds the lower 32 bits of the 68th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address68_high on page 17-1024	0x9A0	32	RW	0xFFFF	Register 616 (MAC Address68 High Register) The MAC Address68 High register holds the upper 16 bits of the 69th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address68 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address68 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address68_low on page 17-1025	0x9A4	32	RW	0xFFFFFFFF	Register 617 (MAC Address68 Low Register) The MAC Address68 Low register holds the lower 32 bits of the 69th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address69_high on page 17-1026	0x9A8	32	RW	0xFFFF	Register 618 (MAC Address69 High Register) The MAC Address69 High register holds the upper 16 bits of the 70th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address69 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address70 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address69_low on page 17-1028	0x9AC	32	RW	0xFFFFFFFF	Register 619 (MAC Address69 Low Register) The MAC Address69 Low register holds the lower 32 bits of the 70th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address70_high on page 17-1028	0x9B0	32	RW	0xFFFF	Register 620 (MAC Address70 High Register) The MAC Address70 High register holds the upper 16 bits of the 71st 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address70 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address70 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address70_low on page 17-1030	0x9B4	32	RW	0xFFFFFFFF	Register 621 (MAC Address70 Low Register) The MAC Address70 Low register holds the lower 32 bits of the 71st 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address71_high on page 17-1031	0x9B8	32	RW	0xFFFF	Register 622 (MAC Address71 High Register) The MAC Address71 High register holds the upper 16 bits of the 72nd 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address71 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address71 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address71_low on page 17-1032	0x9BC	32	RW	0xFFFFFFFF	Register 623 (MAC Address71 Low Register) The MAC Address71 Low register holds the lower 32 bits of the 72nd 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address72_high on page 17-1033	0x9C0	32	RW	0xFFFF	Register 624 (MAC Address72 High Register) The MAC Address72 High register holds the upper 16 bits of the 73rd 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address72 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address72 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address72_low on page 17-1035	0x9C4	32	RW	0xFFFFFFFF	Register 625 (MAC Address72 Low Register) The MAC Address72 Low register holds the lower 32 bits of the 73rd 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address73_high on page 17-1035	0x9C8	32	RW	0xFFFF	Register 626 (MAC Address73 High Register) The MAC Address73 High register holds the upper 16 bits of the 74th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address73 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address73 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address73_low on page 17-1037	0x9CC	32	RW	0xFFFFFFFF	Register 627 (MAC Address73 Low Register) The MAC Address73 Low register holds the lower 32 bits of the 74th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address74_high on page 17-1038	0x9D0	32	RW	0xFFFF	Register 628 (MAC Address74 High Register) The MAC Address74 High register holds the upper 16 bits of the 75th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address74 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address74 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address74_low on page 17-1039	0x9D4	32	RW	0xFFFFFFFF	Register 629 (MAC Address74 Low Register) The MAC Address74 Low register holds the lower 32 bits of the 75th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address75_high on page 17-1040	0x9D8	32	RW	0xFFFF	Register 630 (MAC Address75 High Register) The MAC Address75 High register holds the upper 16 bits of the 76th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address75 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address75 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address75_low on page 17-1042	0x9DC	32	RW	0xFFFFFFFF	Register 631 (MAC Address75 Low Register) The MAC Address75 Low register holds the lower 32 bits of the 76th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address76_high on page 17-1042	0x9E0	32	RW	0xFFFF	Register 632 (MAC Address76 High Register) The MAC Address76 High register holds the upper 16 bits of the 77th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address76 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address76 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address76_low on page 17-1044	0x9E4	32	RW	0xFFFFFFFF	Register 633 (MAC Address76 Low Register) The MAC Address76 Low register holds the lower 32 bits of the 77th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address77_high on page 17-1045	0x9E8	32	RW	0xFFFF	Register 634 (MAC Address77 High Register) The MAC Address77 High register holds the upper 16 bits of the 78th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address77 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address77 Low Register must be performed after at least four clock cycles in the destination clock domain.
gmacgrp_mac_address77_low on page 17-1046	0x9EC	32	RW	0xFFFFFFFF	Register 635 (MAC Address77 Low Register) The MAC Address77 Low register holds the lower 32 bits of the 78th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address78_high on page 17-1047	0x9F0	32	RW	0xFFFF	Register 636 (MAC Address78 High Register) The MAC Address78 High register holds the upper 16 bits of the 79th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address78 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address78 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address78_low on page 17-1049	0x9F4	32	RW	0xFFFFFFFF	Register 637 (MAC Address78 Low Register) The MAC Address78 Low register holds the lower 32 bits of the 79th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address79_high on page 17-1049	0x9F8	32	RW	0xFFFF	Register 638 (MAC Address79 High Register) The MAC Address79 High register holds the upper 16 bits of the 80th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address79 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address79 Low Register must be performed after at least four clock cycles in the destination clock domain.
gmacgrp_mac_address79_low on page 17-1051	0x9FC	32	RW	0xFFFFFFFF	Register 639 (MAC Address79 Low Register) The MAC Address79 Low register holds the lower 32 bits of the 80th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address80_high on page 17-1052	0xA00	32	RW	0xFFFF	Register 640 (MAC Address80 High Register) The MAC Address80 High register holds the upper 16 bits of the 81st 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address80 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address80 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address80_low on page 17-1053	0xA04	32	RW	0xFFFFFFFF	Register 641 (MAC Address80 Low Register) The MAC Address80 Low register holds the lower 32 bits of the 81st 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address81_high on page 17-1054	0xA08	32	RW	0xFFFF	Register 642 (MAC Address81 High Register) The MAC Address81 High register holds the upper 16 bits of the 82nd 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address81 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address81 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address81_low on page 17-1056	0xA0C	32	RW	0xFFFFFFFF	Register 643 (MAC Address81 Low Register) The MAC Address81 Low register holds the lower 32 bits of the 82nd 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address82_high on page 17-1056	0xA10	32	RW	0xFFFF	Register 644 (MAC Address82 High Register) The MAC Address82 High register holds the upper 16 bits of the 83rd 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address82 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address82 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address82_low on page 17-1058	0xA14	32	RW	0xFFFFFFFF	Register 645 (MAC Address82 Low Register) The MAC Address82 Low register holds the lower 32 bits of the 83rd 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address83_high on page 17-1059	0xA18	32	RW	0xFFFF	Register 646 (MAC Address83 High Register) The MAC Address83 High register holds the upper 16 bits of the 84th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address83 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address83 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address83_low on page 17-1060	0xA1C	32	RW	0xFFFFFFFF	Register 647 (MAC Address83 Low Register) The MAC Address83 Low register holds the lower 32 bits of the 84th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address84_high on page 17-1061	0xA20	32	RW	0xFFFF	Register 648 (MAC Address84 High Register) The MAC Address84 High register holds the upper 16 bits of the 85th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address84 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address84 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address84_low on page 17-1063	0xA24	32	RW	0xFFFFFFFF	Register 649 (MAC Address84 Low Register) The MAC Address84 Low register holds the lower 32 bits of the 85th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address85_high on page 17-1063	0xA28	32	RW	0xFFFF	Register 650 (MAC Address85 High Register) The MAC Address85 High register holds the upper 16 bits of the 86th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address85 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address85 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address85_low on page 17-1065	0xA2C	32	RW	0xFFFFFFFF	Register 651 (MAC Address85 Low Register) The MAC Address85 Low register holds the lower 32 bits of the 86th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address86_high on page 17-1066	0xA30	32	RW	0xFFFF	Register 652 (MAC Address86 High Register) The MAC Address86 High register holds the upper 16 bits of the 87th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address86 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address86 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address86_low on page 17-1067	0xA34	32	RW	0xFFFFFFFF	Register 653 (MAC Address86 Low Register) The MAC Address86 Low register holds the lower 32 bits of the 87th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address87_high on page 17-1068	0xA38	32	RW	0xFFFF	Register 654 (MAC Address87 High Register) The MAC Address87 High register holds the upper 16 bits of the 88th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address87 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address87 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address87_low on page 17-1070	0xA3C	32	RW	0xFFFFFFFF	Register 655 (MAC Address87 Low Register) The MAC Address87 Low register holds the lower 32 bits of the 88th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address88_high on page 17-1070	0xA40	32	RW	0xFFFF	Register 656 (MAC Address88 High Register) The MAC Address88 High register holds the upper 16 bits of the 89th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address88 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address88 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address88_low on page 17-1072	0xA44	32	RW	0xFFFFFFFF	Register 657 (MAC Address88 Low Register) The MAC Address88 Low register holds the lower 32 bits of the 89th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address89_high on page 17-1073	0xA48	32	RW	0xFFFF	Register 658 (MAC Address89 High Register) The MAC Address89 High register holds the upper 16 bits of the 90th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address89 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address89 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address89_low on page 17-1074	0xA4C	32	RW	0xFFFFFFFF	Register 659 (MAC Address89 Low Register) The MAC Address89 Low register holds the lower 32 bits of the 90th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address90_high on page 17-1075	0xA50	32	RW	0xFFFF	Register 660 (MAC Address90 High Register) The MAC Address90 High register holds the upper 16 bits of the 91st 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address90 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address90 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address90_low on page 17-1077	0xA54	32	RW	0xFFFFFFFF	Register 661 (MAC Address90 Low Register) The MAC Address90 Low register holds the lower 32 bits of the 91st 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address91_high on page 17-1077	0xA58	32	RW	0xFFFF	Register 662 (MAC Address91 High Register) The MAC Address91 High register holds the upper 16 bits of the 92nd 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address32 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address91 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address91_low on page 17-1079	0xA5C	32	RW	0xFFFFFFFF	Register 663 (MAC Address91 Low Register) The MAC Address91 Low register holds the lower 32 bits of the 92nd 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address92_high on page 17-1080	0xA60	32	RW	0xFFFF	Register 664 (MAC Address92 High Register) The MAC Address92 High register holds the upper 16 bits of the 93rd 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address92 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address92 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address92_low on page 17-1081	0xA64	32	RW	0xFFFFFFFF	Register 665 (MAC Address92 Low Register) The MAC Address92 Low register holds the lower 32 bits of the 93rd 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address93_high on page 17-1082	0xA68	32	RW	0xFFFF	Register 666 (MAC Address93 High Register) The MAC Address93 High register holds the upper 16 bits of the 94th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address93 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address93 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address93_low on page 17-1084	0xA6C	32	RW	0xFFFFFFFF	Register 667 (MAC Address93 Low Register) The MAC Address93 Low register holds the lower 32 bits of the 94th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address94_high on page 17-1084	0xA70	32	RW	0xFFFF	Register 668 (MAC Address94 High Register) The MAC Address94 High register holds the upper 16 bits of the 95th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address94 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address94 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address94_low on page 17-1086	0xA74	32	RW	0xFFFFFFFF	Register 669 (MAC Address94 Low Register) The MAC Address94 Low register holds the lower 32 bits of the 95th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address95_high on page 17-1087	0xA78	32	RW	0xFFFF	Register 670 (MAC Address95 High Register) The MAC Address95 High register holds the upper 16 bits of the 96th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address95 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address95 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address95_low on page 17-1088	0xA7C	32	RW	0xFFFFFFFF	Register 671 (MAC Address95 Low Register) The MAC Address95 Low register holds the lower 32 bits of the 96th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address96_high on page 17-1089	0xA80	32	RW	0xFFFF	Register 672 (MAC Address96 High Register) The MAC Address96 High register holds the upper 16 bits of the 97th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address96 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address96 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address96_low on page 17-1091	0xA84	32	RW	0xFFFFFFFF	Register 673 (MAC Address96 Low Register) The MAC Address96 Low register holds the lower 32 bits of the 97th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address97_high on page 17-1091	0xA88	32	RW	0xFFFF	Register 674 (MAC Address97 High Register) The MAC Address97 High register holds the upper 16 bits of the 98th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address97 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address97 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address97_low on page 17-1093	0xA8C	32	RW	0xFFFFFFFF	Register 675 (MAC Address97 Low Register) The MAC Address97 Low register holds the lower 32 bits of the 98th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address98_high on page 17-1094	0xA90	32	RW	0xFFFF	Register 676 (MAC Address98 High Register) The MAC Address99 High register holds the upper 16 bits of the 100th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address99 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address99 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address98_low on page 17-1095	0xA94	32	RW	0xFFFFFFFF	Register 677 (MAC Address98 Low Register) The MAC Address98 Low register holds the lower 32 bits of the 99th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address99_high on page 17-1096	0xA98	32	RW	0xFFFF	Register 678 (MAC Address99 High Register) The MAC Address99 High register holds the upper 16 bits of the 6-byte 100th MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address99 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address99 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address99_low on page 17-1098	0xA9C	32	RW	0xFFFFFFFF	Register 679 (MAC Address99 Low Register) The MAC Address99 Low register holds the lower 32 bits of the 100th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address100_high on page 17-1098	0xAA0	32	RW	0xFFFF	Register 680 (MAC Address100 High Register) The MAC Address100 High register holds the upper 16 bits of the 101th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address100 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address100 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address100_low on page 17-1100	0xAA4	32	RW	0xFFFFFFFF	Register 681 (MAC Address100 Low Register) The MAC Address100 Low register holds the lower 32 bits of the 101th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address101_high on page 17-1101	0xAA8	32	RW	0xFFFF	Register 682 (MAC Address101 High Register) The MAC Address101 High register holds the upper 16 bits of the 102nd 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address101 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address101 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address101_low on page 17-1102	0xAAC	32	RW	0xFFFFFFFF	Register 683 (MAC Address101 Low Register) The MAC Address101 Low register holds the lower 32 bits of the 102nd 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address102_high on page 17-1103	0xAB0	32	RW	0xFFFF	Register 684 (MAC Address102 High Register) The MAC Address102 High register holds the upper 16 bits of the 6-byte 103rd MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address102 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address102 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address102_low on page 17-1105	0xAB4	32	RW	0xFFFFFFFF	Register 685 (MAC Address102 Low Register) The MAC Address102 Low register holds the lower 32 bits of the 103rd 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address103_high on page 17-1105	0xAB8	32	RW	0xFFFF	Register 686 (MAC Address103 High Register) The MAC Address103 High register holds the upper 16 bits of the 6-byte 104th MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address103 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address103 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address103_low on page 17-1107	0xABC	32	RW	0xFFFFFFFF	Register 687 (MAC Address103 Low Register) The MAC Address103 Low register holds the lower 32 bits of the 104th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address104_high on page 17-1108	0xAC0	32	RW	0xFFFF	Register 688 (MAC Address104 High Register) The MAC Address104 High register holds the upper 16 bits of the 105th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address104 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address104 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address104_low on page 17-1109	0xAC4	32	RW	0xFFFFFFFF	Register 689 (MAC Address104 Low Register) The MAC Address104 Low register holds the lower 32 bits of the 105th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address105_high on page 17-1110	0xAC8	32	RW	0xFFFF	Register 690 (MAC Address105 High Register) The MAC Address105 High register holds the upper 16 bits of the 106th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address105 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address105 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address105_low on page 17-1112	0xACC	32	RW	0xFFFFFFFF	Register 691 (MAC Address105 Low Register) The MAC Address105 Low register holds the lower 32 bits of the 106th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address106_high on page 17-1112	0xAD0	32	RW	0xFFFF	Register 692 (MAC Address106 High Register) The MAC Address106 High register holds the upper 16 bits of the 107th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address106 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address106 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address106_low on page 17-1114	0xAD4	32	RW	0xFFFFFFFF	Register 693 (MAC Address106 Low Register) The MAC Address106 Low register holds the lower 32 bits of the 107th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address107_high on page 17-1115	0xAD8	32	RW	0xFFFF	Register 694 (MAC Address107 High Register) The MAC Address107 High register holds the upper 16 bits of the 108th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address107 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address107 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address107_low on page 17-1116	0xADC	32	RW	0xFFFFFFFF	Register 695 (MAC Address107 Low Register) The MAC Address107 Low register holds the lower 32 bits of the 108th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address108_high on page 17-1117	0xAE0	32	RW	0xFFFF	Register 696 (MAC Address108 High Register) The MAC Address108 High register holds the upper 16 bits of the 109th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address108 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address108 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address108_low on page 17-1119	0xAE4	32	RW	0xFFFFFFFF	Register 697 (MAC Address108 Low Register) The MAC Address108 Low register holds the lower 32 bits of the 109th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address109_high on page 17-1119	0xAE8	32	RW	0xFFFF	Register 698 (MAC Address109 High Register) The MAC Address109 High register holds the upper 16 bits of the 110th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address109 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address109 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address109_low on page 17-1121	0xAEC	32	RW	0xFFFFFFFF	Register 699 (MAC Address109 Low Register) The MAC Address109 Low register holds the lower 32 bits of the 110th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address110_high on page 17-1122	0xAF0	32	RW	0xFFFF	Register XXX (MAC AddressXX High Register) The MAC Address110 High register holds the upper 16 bits of the 111th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address110 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address110 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address110_low on page 17-1123	0xAF4	32	RW	0xFFFFFFFF	Register 700 (MAC Address110 Low Register) The MAC Address110 Low register holds the lower 32 bits of the 111th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address111_high on page 17-1124	0xAF8	32	RW	0xFFFF	Register 701 (MAC Address111 High Register) The MAC Address111 High register holds the upper 16 bits of the 6-byte 112th MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address111 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address111 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address111_low on page 17-1126	0xAFC	32	RW	0xFFFFFFFF	Register 702 (MAC Address111 Low Register) The MAC Address111 Low register holds the lower 32 bits of the 112th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address112_high on page 17-1126	0xB00	32	RW	0xFFFF	Register 703 (MAC Address112 High Register) The MAC Address112 High register holds the upper 16 bits of the 113th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address112 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address112 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address112_low on page 17-1128	0xB04	32	RW	0xFFFFFFFF	Register 704 (MAC Address112 Low Register) The MAC Address112 Low register holds the lower 32 bits of the 113th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address113_high on page 17-1129	0xB08	32	RW	0xFFFF	Register 705 (MAC Address113 High Register) The MAC Address113 High register holds the upper 16 bits of the 114th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address113 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address113 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address113_low on page 17-1130	0xB0C	32	RW	0xFFFFFFFF	Register 706 (MAC Address113 Low Register) The MAC Address113 Low register holds the lower 32 bits of the 114th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address114_high on page 17-1131	0xB10	32	RW	0xFFFF	Register 707 (MAC Address114 High Register) The MAC Address114 High register holds the upper 16 bits of the 115th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address114 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address114 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address114_low on page 17-1133	0xB14	32	RW	0xFFFFFFFF	Register 708 (MAC Address114 Low Register) The MAC Address114 Low register holds the lower 32 bits of the 115th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address115_high on page 17-1133	0xB18	32	RW	0xFFFF	Register 709 (MAC Address115 High Register) The MAC Address115 High register holds the upper 16 bits of the 116th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address115 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address115 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address115_low on page 17-1135	0xB1C	32	RW	0xFFFFFFFF	Register 710 (MAC Address115 Low Register) The MAC Address115 Low register holds the lower 32 bits of the 116th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address116_high on page 17-1136	0xB20	32	RW	0xFFFF	Register 711 (MAC Address116 High Register) The MAC Address116 High register holds the upper 16 bits of the 117th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address116 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address116 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address116_low on page 17-1137	0xB24	32	RW	0xFFFFFFFF	Register 712 (MAC Address116 Low Register) The MAC Address116 Low register holds the lower 32 bits of the 117th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address117_high on page 17-1138	0xB28	32	RW	0xFFFF	Register 713 (MAC Address117 High Register) The MAC Address117 High register holds the upper 16 bits of the 118th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode of the MAC Address117 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address117 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address117_low on page 17-1140	0xB2C	32	RW	0xFFFFFFFF	Register 714 (MAC Address117 Low Register) The MAC Address117 Low register holds the lower 32 bits of the 118th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address118_high on page 17-1140	0xB30	32	RW	0xFFFF	Register 715 (MAC Address118 High Register) The MAC Address118 High register holds the upper 16 bits of the 119th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address118 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address118 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address118_low on page 17-1142	0xB34	32	RW	0xFFFFFFFF	Register 716 (MAC Address118 Low Register) The MAC Address118 Low register holds the lower 32 bits of the 119th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address119_high on page 17-1143	0xB38	32	RW	0xFFFF	Register 717 (MAC Address119 High Register) The MAC Address119 High register holds the upper 16 bits of the 120th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address119 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address119 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address119_low on page 17-1144	0xB3C	32	RW	0xFFFFFFFF	Register 718 (MAC Address119 Low Register) The MAC Address119 Low register holds the lower 32 bits of the 120th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address120_high on page 17-1145	0xB40	32	RW	0xFFFF	Register 719 (MAC Address120 High Register) The MAC Address120 High register holds the upper 16 bits of the 6-byte 121st MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address120 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address120 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address120_low on page 17-1147	0xB44	32	RW	0xFFFFFFFF	Register 720 (MAC Address120 Low Register) The MAC Address120 Low register holds the lower 32 bits of the 121st 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address121_high on page 17-1147	0xB48	32	RW	0xFFFF	Register 721 (MAC Address121 High Register) The MAC Address121 High register holds the upper 16 bits of the 122nd 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address121 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address121 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address121_low on page 17-1149	0xB4C	32	RW	0xFFFFFFFF	Register 722 (MAC Address121 Low Register) The MAC Address121 Low register holds the lower 32 bits of the 122nd 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address122_high on page 17-1150	0xB50	32	RW	0xFFFF	Register 723 (MAC Address122 High Register) The MAC Address122 High register holds the upper 16 bits of the 123rd 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address122 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address122 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address122_low on page 17-1151	0xB54	32	RW	0xFFFFFFFF	Register 724 (MAC Address122 Low Register) The MAC Address122 Low register holds the lower 32 bits of the 123rd 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address123_high on page 17-1152	0xB58	32	RW	0xFFFF	Register 725 (MAC Address123 High Register) The MAC Address123 High register holds the upper 16 bits of the 124th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address123 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address123 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address123_low on page 17-1154	0xB5C	32	RW	0xFFFFFFFF	Register 726 (MAC AddressXX 123 Register) The MAC Address123 Low register holds the lower 32 bits of the 124th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address124_high on page 17-1154	0xB60	32	RW	0xFFFF	Register 727 (MAC Address124 High Register) The MAC Address124 High register holds the upper 16 bits of the 125th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address124 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address124 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address124_low on page 17-1156	0xB64	32	RW	0xFFFFFFFF	Register 728 (MAC Address124 Low Register) The MAC Address124 Low register holds the lower 32 bits of the 125th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address125_high on page 17-1157	0xB68	32	RW	0xFFFF	Register 729 (MAC Address125 High Register) The MAC Address125 High register holds the upper 16 bits of the 126th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address125 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address125 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address125_low on page 17-1158	0xB6C	32	RW	0xFFFFFFFF	Register 730 (MAC Address125 Low Register) The MAC Address125 Low register holds the lower 32 bits of the 126th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address126_high on page 17-1159	0xB70	32	RW	0xFFFF	Register 731 (MAC Address126 High Register) The MAC Address126 High register holds the upper 16 bits of the 127th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address126 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address126 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address126_low on page 17-1161	0xB74	32	RW	0xFFFFFFFF	Register 732 (MAC Address126 Low Register) The MAC Address126 Low register holds the lower 32 bits of the 127th 6-byte MAC address of the station.

Register	Offset	Width	Access	Reset Value	Description
gmacgrp_mac_address127_high on page 17-1161	0xB78	32	RW	0xFFFF	Register 733 (MAC Address127 High Register) The MAC Address127 High register holds the upper 16 bits of the 128th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address127 Low Register are written. For proper synchroniza-tion updates, consecu-tive writes to this MAC Address127 Low Register must be performed after at least four clock cycles in the destina-tion clock domain.
gmacgrp_mac_address127_low on page 17-1163	0xB7C	32	RW	0xFFFFFFFF	Register 734 (MAC Address127 Low Register) The MAC Address127 Low register holds the lower 32 bits of the 128th 6-byte MAC address of the station.
dmagrp_bus_mode on page 17-1164	0x1000	32	RW	0x20101	Register 0 (Bus Mode Register) The Bus Mode register establishes the bus operating modes for the DMA.

Register	Offset	Width	Access	Reset Value	Description
dmagrp_transmit_poll_demand on page 17-1171	0x1004	32	RW	0x0	Register 1 (Transmit Poll Demand Register) The Transmit Poll Demand register enables the Tx DMA to check whether or not the DMA owns the current descriptor. The Transmit Poll Demand command is given to wake up the Tx DMA if it is in the Suspend mode. The Tx DMA can go into the Suspend mode because of an Underflow error in a transmitted frame or the unavailability of descriptors owned by it. You can give this command anytime and the Tx DMA resets this command when it again starts fetching the current descriptor from host memory. When this register is read, it always returns zero.
dmagrp_receive_poll_demand on page 17-1172	0x1008	32	RW	0x0	Register 2 (Receive Poll Demand Register) The Receive Poll Demand register enables the receive DMA to check for new descriptors. This command is used to wake up the Rx DMA from the SUSPEND state. The RxDMA can go into the SUSPEND state only because of the unavailability of descriptors it owns. When this register is read, it always returns zero.

Register	Offset	Width	Access	Reset Value	Description
dmagrp_receive_descriptor_list_address on page 17-1173	0x100C	32	RW	0x0	Register 3 (Receive Descriptor List Address Register) The Receive Descriptor List Address register points to the start of the Receive Descriptor List. The descriptor lists reside in the host's physical memory space and must be Word, Dword, or Lword-aligned (for 32-bit, 64-bit, or 128-bit data bus). The DMA internally converts it to bus width aligned address by making the corresponding LS bits low. Writing to this register is permitted only when reception is stopped. When stopped, this register must be written to before the receive Start command is given. You can write to this register only when Rx DMA has stopped, that is, Bit 1 (SR) is set to zero in Register 6 (Operation Mode Register). When stopped, this register can be written with a new descriptor list address. When you set the SR bit to 1, the DMA takes the newly programmed descriptor base address. If this register is not changed when the SR bit is set to 0, then the DMA takes the descriptor address where it was stopped earlier.

Register	Offset	Width	Access	Reset Value	Description
dmagrp_transmit_descriptor_list_address on page 17-1175	0x1010	32	RW	0x0	Register 4 (Transmit Descriptor List Address Register) The Transmit Descriptor List Address register points to the start of the Transmit Descriptor List. The descriptor lists reside in the host's physical memory space and must be Word, Dword, or Lword-aligned (for 32-bit, 64-bit, or 128-bit data bus). The DMA internally converts it to bus width aligned address by making the corresponding LSB to low. You can write to this register only when the Tx DMA has stopped, that is, Bit 13 (ST) is set to zero in Register 6 (Operation Mode Register). When stopped, this register can be written with a new descriptor list address. When you set the ST bit to 1, the DMA takes the newly programmed descriptor base address. If this register is not changed when the ST bit is set to 0, then the DMA takes the descriptor address where it was stopped earlier.

Register	Offset	Width	Access	Reset Value	Description
dmagrp_status on page 17-1176	0x1014	32	RW	0x0	Register 5 (Status Register) The Status register contains all status bits that the DMA reports to the host. The Software driver reads this register during an interrupt service routine or polling. Most of the fields in this register cause the host to be interrupted. The bits of this register are not cleared when read. Writing 1'b1 to (unreserved) Bits[16:0] of this register clears these bits and writing 1'b0 has no effect. Each field (Bits[16:0]) can be masked by masking the appropriate bit in Register 7 (Interrupt Enable Register).
dmagrp_operation_mode on page 17-1188	0x1018	32	RW	0x0	Register 6 (Operation Mode Register) The Operation Mode register establishes the Transmit and Receive operating modes and commands. This register should be the last CSR to be written as part of the DMA initialization. This register is also present in the GMAC-MTL configuration with unused and reserved bits 24, 13, 2, and 1.

Register	Offset	Width	Access	Reset Value	Description
dmagrp_interrupt_enable on page 17-1200	0x101C	32	RW	0x0	Register 7 (Interrupt Enable Register) The Interrupt Enable register enables the interrupts reported by Register 5 (Status Register). Setting a bit to 1'b1 enables a corresponding interrupt. After a hardware or software reset, all interrupts are disabled.
dmagrp_missed_frame_and_buffer_overflow_counter on page 17-1206	0x1020	32	RO	0x0	Register 8 (Missed Frame and Buffer Overflow Counter Register) The DMA maintains two counters to track the number of frames missed during reception. This register reports the current value of the counter. The counter is used for diagnostic purposes. Bits[15:0] indicate missed frames because of the host buffer being unavail-able. Bits[27:17] indicate missed frames because of buffer overflow conditions (MTL and MAC) and runt frames (good frames of less than 64 bytes) dropped by the MTL.

Register	Offset	Width	Access	Reset Value	Description
dmagrp_receive_interrupt_watchdog_timer on page 17-1208	0x1024	32	RW	0x0	Register 9 (Receive Interrupt Watchdog Timer Register) This register, when written with non-zero value, enables the watchdog timer for the Receive Interrupt (Bit 6) of Register 5 (Status Register)
dmagrp_axi_bus_mode on page 17-1210	0x1028	32	RW	0x110001	Register 10 (AXI Bus Mode Register)
dmagrp_ahb_or_axi_status on page 17-1214	0x102C	32	RO	0x0	Register 11 (AHB or AXI Status Register) This register provides the active status of the AHB master interface or AXI interface's read and write channels. This register is present and valid only in the GMAC-AHB and GMAC-AXI configurations. This register is useful for debugging purposes. In addition, this register is valid only in the Channel 0 DMA when multiple channels are present in the AV mode.
dmagrp_current_host_transmit_descriptor on page 17-1215	0x1048	32	RO	0x0	Register 18 (Current Host Transmit Descriptor Register) The Current Host Transmit Descriptor register points to the start address of the current Transmit Descriptor read by the DMA.

Register	Offset	Width	Access	Reset Value	Description
dmagrp_current_host_receive_descriptor on page 17-1216	0x104C	32	RO	0x0	Register 19 (Current Host Receive Descriptor Register) The Current Host Receive Descriptor register points to the start address of the current Receive Descriptor read by the DMA.
dmagrp_current_host_transmit_buffer_address on page 17-1217	0x1050	32	RO	0x0	Register 20 (Current Host Transmit Buffer Address Register) The Current Host Transmit Buffer Address register points to the current Transmit Buffer Address being read by the DMA.
dmagrp_current_host_receive_buffer_address on page 17-1218	0x1054	32	RO	0x0	Register 21 (Current Host Receive Buffer Address Register) The Current Host Receive Buffer Address register points to the current Receive Buffer address being read by the DMA.

Register	Offset	Width	Access	Reset Value	Description
dmagrp_hw_feature on page 17-1218	0x1058	32	RO	0xF0D69BF	Register 22 (HW Feature Register) This register indicates the presence of the optional features or functions of the DWC_gmac. The software driver can use this register to dynamically enable or disable the programs related to the optional blocks. Note: All bits are set or reset as per the selection of features during the DWC_gmac configuration.

emac Summary

Module Instance	Base Address
i_emac_emac0	0xFF800000
i_emac_emac1	0xFF802000
i_emac_emac2	0xFF804000

Register Address Offset	Bit Fields															
i_emac_emac0																
gmacgrp_mac_configuration 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_28 RO 0x0				twokpe RW 0x0	sfterr RO 0x0	cst RW 0x0	tc RO 0x0	wd RW 0x0	jd RW 0x0	be RW 0x0	je RW 0x0	ifg RW 0x0			dcrs RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ps RW 0x0	fes RW 0x0	do RW 0x0	lm RW 0x0	dm RW 0x0	ipc RW 0x0	dr RW 0x0	lud RO 0x0	acs RW 0x0	bl RW 0x0		dc RW 0x0	te RW 0x0	re RW 0x0	prelen RW 0x0	

Register Address Offset	Bit Fields															
gmacgrp_mac_frame_filter 0x4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ra RW 0x0	reserved_30_22 RO 0x0									dntu RW 0x0	ipfe RW 0x0	reserved_19_17 RO 0x0			vtfe RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved_15_11 RO 0x0					hpf RO 0x0	saf RW 0x0	saif RW 0x0	pcf RW 0x0		dbf RW 0x0	pm RW 0x0	daif RW 0x0	hmc RO 0x0	huc RO 0x0	pr RW 0x0
gmacgrp_gmii_address 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_16 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	pa RW 0x0					gr RW 0x0					cr RW 0x0				gw RW 0x0	gb RW 0x0
gmacgrp_gmii_data 0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_16 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	gd RW 0x0															
gmacgrp_flow_control 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	pt RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved_15_8 RO 0x0								dzpq RW 0x0	reserved_6 RO 0x0	plt RW 0x0		up RW 0x0	rfe RW 0x0	tfe RW 0x0	fca_bpa RW 0x0
gmacgrp_vlan_tag 0x1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_20 RO 0x0												vthm RO 0x0	esvl RW 0x0	vtim RW 0x0	etv RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	vl RW 0x0															

Register Address Offset	Bit Fields															
gmacgrp_version 0x20	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_16 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	userver RO 0x10								snpsver RO 0x37							
gmacgrp_debug 0x24	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_26 RO 0x0						txstsfsts RO 0x0	txfsts RO 0x0	reserved_23 RO 0x0	twcsts RO 0x0	trcsts RO 0x0		txpaused RO 0x0	tfcsts RO 0x0		tpests RO 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved_15_10 RO 0x0						rxfsts RO 0x0		reserved_7 RO 0x0	rrcsts RO 0x0		rwcsts RO 0x0	reserved_3 RO 0x0	rfcfcsts RO 0x0		rpests RO 0x0
gmacgrp_lpi_control_status 0x30	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_20 RO 0x0												lpitxa RW 0x0	plsen RO 0x0	pls RW 0x0	lpien RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved_15_10 RO 0x0						rlpist RO 0x0	tlpist RO 0x0	reserved_7_4 RO 0x0				rlpiex RO 0x0	rlpien RO 0x0	tlpiex RO 0x0	tlpien RO 0x0
gmacgrp_lpi_timers_control 0x34	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_26 RO 0x0						lst RW 0x3E8									
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	twt RW 0x0															
gmacgrp_interrupt_status 0x38	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_12 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved_31_12 RO 0x0				gpiis RO 0x0	lpiis RO 0x0	tsis RO 0x0	reserved_8 RO 0x0	mmcrxipis RO 0x0	mmctxis RO 0x0	mmcrxis RO 0x0	mmcis RO 0x0	pmtis RO 0x0	pcsancis RO 0x0	pcslchgis RO 0x0	rgsmiiis RO 0x0

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_11 RO 0x0															
gmacgrp_interrupt_mask 0x3C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved_31_11 RO 0x0					lpiim RW 0x0	tsim RO 0x0	reserved_8_4 RO 0x0					pmtim RW 0x0	pcsancim RO 0x0	pcslchgim RO 0x0	rgsmiiim RO 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gmacgrp_mac_address0_high 0x40	ae RO 0x1	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gmacgrp_mac_address0_low 0x44	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gmacgrp_mac_address1_high 0x48	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gmacgrp_mac_address1_low 0x4C	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address2_high 0x50	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address2_low 0x54	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address3_high 0x58	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address3_low 0x5C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address4_high 0x60	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address4_low 0x64	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address5_high 0x68	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address5_low 0x6C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address6_high 0x70	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address6_low 0x74	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address7_high 0x78	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address7_low 0x7C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address8_high 0x80	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address8_low 0x84	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address9_high 0x88	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gmacgrp_mac_address9_low 0x8C	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address10_high 0x90	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address10_low 0x94	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address11_high 0x98	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address11_low 0x9C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address12_high 0xA0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address12_low 0xA4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address13_high 0xA8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address13_low 0xAC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address14_high 0xB0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address14_low 0xB4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address15_high 0xB8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address15_low 0xBC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_sgmii_rgmii_smii_control_status 0xD8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved												lnksts RO 0x0	lnkspeed RO 0x0		lnkmod RO 0x0
gmacgrp_wdog_timeout 0xDC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_17 RO 0x0															pwe RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved_15_14 RO 0x0		wto RW 0x0													

Register Address Offset	Bit Fields															
gmacgrp_gen pio 0xE0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_x RO 0x0							gpit RW 0x0	reserved_23_x RO 0x0							gpie RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved_15_x RO 0x0							gpo RW 0x0	reserved_7_x RO 0x0							gpis RO 0x0
gmacgrp_mmc _control 0x100	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_9 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved_31_9 RO 0x0							ucdbc RW 0x0	reserved_7_6 RO 0x0		cntprstlvl RW 0x0	cntprst RW 0x0	cntfreez RW 0x0	rstonrd RW 0x0	cntstopro RW 0x0	cntrst RW 0x0
gmacgrp_mmc _receive_in terrupt 0x104	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_26 RO 0x0						rxctrlfis RO 0x0	rxrcverrfis RO 0x0	rxwdogfis RO 0x0	rxvlangbfis RO 0x0	rxfovfis RO 0x0	rxpausfis RO 0x0	rxorangefis RO 0x0	rxlenerfis RO 0x0	rxucgfis RO 0x0	rx1024tmaxoctgbfis RO 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rx512t1023octgbfis RO 0x0	rx256t511octgbfis RO 0x0	rx128t255octgbfis RO 0x0	rx65t127octgbfis RO 0x0	rx64octgbfis RO 0x0	rxosizegfis RO 0x0	rxusizegfis RO 0x0	rxjaberfis RO 0x0	rxruntfis RO 0x0	rxalignerfis RO 0x0	rxcrcerfis RO 0x0	rxmcgfis RO 0x0	rxbcgfis RO 0x0	rxgoctis RO 0x0	rxgboctis RO 0x0	rxgbfrmis RO 0x0
gmacgrp_mmc _transmit_i nterrupt 0x108	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_26 RO 0x0						txosizegfis RO 0x0	txvlangfis RO 0x0	txpausfis RO 0x0	txexdeffis RO 0x0	txgfrmis RO 0x0	txgoctis RO 0x0	txcarerfis RO 0x0	txexcolfis RO 0x0	txlatcolfis RO 0x0	txdeffis RO 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	txmcolgfis RO 0x0	txscolgfis RO 0x0	txuflowerfis RO 0x0	txbcgbfis RO 0x0	txmcgbfis RO 0x0	txucgbfis RO 0x0	tx1024tmaxoctgbfis RO 0x0	tx512t1023octgbfis RO 0x0	tx256t511octgbfis RO 0x0	tx128t255octgbfis RO 0x0	tx65t127octgbfis RO 0x0	tx64octgbfis RO 0x0	txmcgfis RO 0x0	txbcgfis RO 0x0	txgbfrmis RO 0x0	txgboctis RO 0x0

Register Address Offset	Bit Fields															
gmacgrp_mmc_receive_interrupt_mask 0x10C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_26 RO 0x0						rxctrlfim RW 0x0	rxrcverrfim RW 0x0	rxwdogfim RW 0x0	rxvlangbfim RW 0x0	rxfovfim RW 0x0	rxpausfim RW 0x0	rxorangefim RW 0x0	rxlenerfim RW 0x0	rxucgfim RW 0x0	rx1024tmaxoctgbfim RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rx512t1023octgbfim RW 0x0	rx256t511octgbfim RW 0x0	rx128t255octgbfim RW 0x0	rx65t127octgbfim RW 0x0	rx64octgbfim RW 0x0	rxosizegfim RW 0x0	rxusizegfim RW 0x0	rxjaberfim RW 0x0	rxruntfim RW 0x0	rxalignerfim RW 0x0	rxcrcerfim RW 0x0	rxmcgfim RW 0x0	rxbcgfim RW 0x0	rxgoctim RW 0x0	rxgboctim RW 0x0	rxgbfrmim RW 0x0
gmacgrp_mmc_transmit_interrupt_mask 0x110	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_26 RO 0x0						txosizegfim RW 0x0	txvlangfim RW 0x0	txpausfim RW 0x0	txexdeffim RW 0x0	txgfrmim RW 0x0	txgoctim RW 0x0	txcarerfim RW 0x0	txexcolfim RW 0x0	txlatcolfim RW 0x0	txdeffim RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	txmcolgfim RW 0x0	txscolgfim RW 0x0	txuflowerfim RW 0x0	txbcgbfim RW 0x0	txmcgbfim RW 0x0	txucgbfim RW 0x0	tx1024tmaxoctgbfim RW 0x0	tx512t1023octgbfim RW 0x0	tx256t511octgbfim RW 0x0	tx128t255octgbfim RW 0x0	tx65t127octgbfim RW 0x0	tx64octgbfim RW 0x0	txmcgfim RW 0x0	txbcgfim RW 0x0	txgbfrmim RW 0x0	txgboctim RW 0x0
gmacgrp_txoctetcount_gb 0x114	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_txframecount_gb 0x118	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															

Register Address Offset	Bit Fields															
gmacgrp_txbroadcast-frames_g 0x11C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_txmulticast-frames_g 0x120	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_tx64octets_gb 0x124	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_tx65to127octets_gb 0x128	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_tx128to255octets_gb 0x12C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_tx256to511octets_gb 0x130	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															

Register Address Offset	Bit Fields															
gmacgrp_tx512to1023octets_gb 0x134	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_tx1024tomaxoctets_gb 0x138	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_txunicastframes_gb 0x13C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_txmulticast-frames_gb 0x140	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_txbroadcast-frames_gb 0x144	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_txunderflo-werror 0x148	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															

Register Address Offset	Bit Fields															
gmacgrp_txsinglecol_g 0x14C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_txmulticol_g 0x150	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_txdeferred 0x154	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_txlatecol 0x158	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_txexesscol 0x15C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_txcarriererr 0x160	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															

Register Address Offset	Bit Fields															
gmacgrp_txoctetcnt 0x164	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	txoctetcount_g RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	txoctetcount_g RO 0x0															
gmacgrp_txframecount_g 0x168	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_txexcessdef 0x16C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_txpauseframes 0x170	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_txvlanframes_g 0x174	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_txoversize_g 0x178	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															

Register Address Offset	Bit Fields															
gmacgrp_rxframecount_gb 0x180	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxoctetcount_gb 0x184	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxoctetcount_g 0x188	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxbroadcast-frames_g 0x18C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxmulticast-frames_g 0x190	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxcrcerror 0x194	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															

Register Address Offset	Bit Fields															
gmacgrp_rxalignmenterror 0x198	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxrunterror 0x19C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxjabbererror 0x1A0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxundersize_g 0x1A4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxoversize_g 0x1A8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rx64octets_gb 0x1AC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															

Register Address Offset	Bit Fields															
gmacgrp_rx65to127octets_gb 0x1B0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rx128to255octets_gb 0x1B4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rx256to511octets_gb 0x1B8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rx512to1023octets_gb 0x1BC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rx1024tomaxoctets_gb 0x1C0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxunicastframes_g 0x1C4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															

Register Address Offset	Bit Fields															
gmacgrp_rxlengtherror 0x1C8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxoutofrange-type 0x1CC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxpauseframes 0x1D0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxfifooverflow 0x1D4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxvlanframes_gb 0x1D8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxwatchdogerror 0x1DC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															

Register Address Offset	Bit Fields															
gmacgrp_rxrcverror 0x1E0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxctrlframes_g 0x1E4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_mmc_ipc_receive_interrupt_mask 0x200	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved		rxicmperoim RW 0x0	rxicmpgoim RW 0x0	rxtcperoim RW 0x0	rxtcpgoim RW 0x0	rxudperoim RW 0x0	rxudpgoim RW 0x0	rxipv6nopayoim RW 0x0	rxipv6heroim RW 0x0	rxipv6goim RW 0x0	rxipv4udsbloim RW 0x0	rxipv4fragoim RW 0x0	rxipv4nopayoim RW 0x0	rxipv4heroim RW 0x0	rxipv4goim RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		rxicmperfim RW 0x0	rxicmpgfim RW 0x0	rxtcperfim RW 0x0	rxtcpgfim RW 0x0	rxudperfim RW 0x0	rxudpgfim RW 0x0	rxipv6nopayfim RW 0x0	rxipv6herfim RW 0x0	rxipv6gfim RW 0x0	rxipv4udsblfim RW 0x0	rxipv4fragfim RW 0x0	rxipv4nopayfim RW 0x0	rxipv4herfim RW 0x0	rxipv4gfim RW 0x0
gmacgrp_mmc_ipc_receive_interrupt 0x208	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved		rxicmperois RO 0x0	rxicmpgois RO 0x0	rxtcperois RO 0x0	rxtcpgois RO 0x0	rxudperois RO 0x0	rxudpgois RO 0x0	rxipv6nopayois RO 0x0	rxipv6herois RO 0x0	rxipv6gois RO 0x0	rxipv4udsblois RO 0x0	rxipv4fragois RO 0x0	rxipv4nopayois RO 0x0	rxipv4herois RO 0x0	rxipv4gois RO 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		rxicmperfis RO 0x0	rxicmpgfis RO 0x0	rxtcperfis RO 0x0	rxtcpgfis RO 0x0	rxudperfis RO 0x0	rxudpgfis RO 0x0	rxipv6nopayfis RO 0x0	rxipv6herfis RO 0x0	rxipv6gfis RO 0x0	rxipv4udsblfis RO 0x0	rxipv4fragfis RO 0x0	rxipv4nopayfis RO 0x0	rxipv4herfis RO 0x0	rxipv4gfis RO 0x0

Register Address Offset	Bit Fields															
gmacgrp_rxipv4_gd_frms 0x210	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxipv4_hdrerr_frms 0x214	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxipv4_nopay_frms 0x218	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxipv4_frag_frms 0x21C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxipv4_udsbl_frms 0x220	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxipv6_gd_frms 0x224	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															

Register Address Offset	Bit Fields															
gmacgrp_rxipv6_hdrerr_frms 0x228	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxipv6_nopay_frms 0x22C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxudp_gd_frms 0x230	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxudp_err_frms 0x234	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxtcp_gd_frms 0x238	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxtcp_err_frms 0x23C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															

Register Address Offset	Bit Fields															
gmacgrp_rxicmp_gd_frms 0x240	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxicmp_err_frms 0x244	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxipv4_gd_octets 0x250	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxipv4_hdrerr_octets 0x254	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxipv4_nopay_octets 0x258	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxipv4_frag_octets 0x25C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															

Register Address Offset	Bit Fields															
gmacgrp_rxipv4_udsbl_octets 0x260	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxipv6_gd_octets 0x264	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxipv6_hdrerr_octets 0x268	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxipv6_nopay_octets 0x26C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxudp_gd_octets 0x270	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxudp_err_octets 0x274	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															

Register Address Offset	Bit Fields															
gmacgrp_rxtcp_gd_octets 0x278	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxtcperroctets 0x27C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	rxtcp_err_octets RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rxtcp_err_octets RO 0x0															
gmacgrp_rxicmp_gd_octets 0x280	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxicmp_err_octets 0x284	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_l3_l4_control0 0x400	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_22 RO 0x0										l4dpim0 RW 0x0	l4dpm0 RW 0x0	l4spim0 RW 0x0	l4spm0 RW 0x0	reserved_17 RO 0x0	l4pen0 RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l3hdbm0 RW 0x0					l3hsbm0 RW 0x0					l3daim0 RW 0x0	l3dam0 RW 0x0	l3saim0 RW 0x0	l3sam0 RW 0x0	reserved_1 RO 0x0	l3pen0 RW 0x0

Register Address Offset	Bit Fields															
gmacgrp_layer4_address0 0x404	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	l4dp0 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l4sp0 RW 0x0															
gmacgrp_layer3_addr0_reg0 0x410	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	l3a00 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l3a00 RW 0x0															
gmacgrp_layer3_addr1_reg0 0x414	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	l3a10 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l3a10 RW 0x0															
gmacgrp_layer3_addr2_reg0 0x418	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	l3a20 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l3a20 RW 0x0															
gmacgrp_layer3_addr3_reg0 0x41C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	l3a30 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l3a30 RW 0x0															

Register Address Offset	Bit Fields															
gmacgrp_l3_l4_control1 0x430	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved										l4dpim1 RW 0x0	l4dpm1 RW 0x0	l4spim1 RW 0x0	l4spm1 RW 0x0	Reserved	l4pen1 RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l3hdbm1 RW 0x0					l3hsbm1 RW 0x0					l3daim1 RW 0x0	l3dam1 RW 0x0	l3saim1 RW 0x0	l3sam1 RW 0x0	Reserved	l3pen1 RW 0x0
gmacgrp_layer4_address1 0x434	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	l4dp1 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l4sp1 RW 0x0															
gmacgrp_layer3_addr0_reg1 0x440	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	l3a01 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l3a01 RW 0x0															
gmacgrp_layer3_addr1_reg1 0x444	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	l3a11 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l3a11 RW 0x0															
gmacgrp_layer3_addr2_reg1 0x448	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	l3a21 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l3a21 RW 0x0															

Register Address Offset	Bit Fields															
gmacgrp_layer3_addr3_reg1 0x44C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	l3a31 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l3a31 RW 0x0															
gmacgrp_l3_l4_control2 0x460	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved										l4dpim2 RW 0x0	l4dpm2 RW 0x0	l4spim2 RW 0x0	l4spm2 RW 0x0	Reserved	l4pen2 RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l3hdbm2 RW 0x0					l3hsbm2 RW 0x0					l3daim2 RW 0x0	l3dam2 RW 0x0	l3saim2 RW 0x0	l3sam2 RW 0x0	Reserved	l3pen2 RW 0x0
gmacgrp_layer4_address2 0x464	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	l4dp2 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l4sp2 RW 0x0															
gmacgrp_layer3_addr0_reg2 0x470	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	l3a02 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l3a02 RW 0x0															
gmacgrp_layer3_addr1_reg2 0x474	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	l3a12 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l3a12 RW 0x0															

Register Address Offset	Bit Fields															
gmacgrp_layer3_addr2_reg2 0x478	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	l3a22 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l3a22 RW 0x0															
gmacgrp_layer3_addr3_reg2 0x47C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	l3a32 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l3a32 RW 0x0															
gmacgrp_l3_l4_control3 0x490	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved										l4dpim3 RW 0x0	l4dpm3 RW 0x0	l4spim3 RW 0x0	l4spm3 RW 0x0	Reserved	l4pen3 RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l3hdbm3 RW 0x0					l3hsbm3 RW 0x0					l3daim3 RW 0x0	l3dam3 RW 0x0	l3saim3 RW 0x0	l3sam3 RW 0x0	Reserved	l3pen3 RW 0x0
gmacgrp_layer4_address3 0x494	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	l4dp3 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l4sp3 RW 0x0															
gmacgrp_layer3_addr0_reg3 0x4A0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	l3a03 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l3a03 RW 0x0															

Register Address Offset	Bit Fields															
gmacgrp_layer3_addr1_reg3 0x4A4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	l3a13 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l3a13 RW 0x0															
gmacgrp_layer3_addr2_reg3 0x4A8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	l3a23 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l3a23 RW 0x0															
gmacgrp_layer3_addr3_reg3 0x4AC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	l3a33 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l3a33 RW 0x0															
gmacgrp_hash_table_reg0 0x500	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ht31t0 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ht31t0 RW 0x0															
gmacgrp_hash_table_reg1 0x504	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ht63t32 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ht63t32 RW 0x0															
gmacgrp_hash_table_reg2 0x508	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ht95t64 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ht95t64 RW 0x0															

Register Address Offset	Bit Fields															
gmacgrp_hash_table_reg3 0x50C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ht127t96 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ht127t96 RW 0x0															
gmacgrp_hash_table_reg4 0x510	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ht159t128 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ht159t128 RW 0x0															
gmacgrp_hash_table_reg5 0x514	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ht191t160 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ht191t160 RW 0x0															
gmacgrp_hash_table_reg6 0x518	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ht223t196 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ht223t196 RW 0x0															
gmacgrp_hash_table_reg7 0x51C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ht255t224 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ht255t224 RW 0x0															
gmacgrp_vlan_incl_reg 0x584	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_20 RO 0x0												csvl RW 0x0	vlp RW 0x0	vlc RW 0x0	
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	vlt RW 0x0															

Register Address Offset	Bit Fields															
gmacgrp_vlan_hash_table_reg 0x588	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	vlht RW 0x0															
gmacgrp_timestamp_control 0x700	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_29 RO 0x0			atsen3 RO 0x0	atsen2 RO 0x0	atsen1 RO 0x0	atsen0 RO 0x0	atsfc RO 0x0	reserved_23_19 RO 0x0					tsenmacaddr RW 0x0	snaptypsel RW 0x0	
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	tsmstrena RW 0x0	tsevntena RW 0x0	tsipv4ena RW 0x1	tsipv6ena RW 0x0	tsipena RW 0x0	tsver2ena RW 0x0	tsctrlssr RW 0x0	tsenall RW 0x0	reserved_7_6 RO 0x0		tsaddreg RO 0x0	tstrig RO 0x0	tsupdt RO 0x0	tsinit RO 0x0	tscfupdt RO 0x0	tsena RW 0x0
gmacgrp_sub_second_increment 0x704	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								ssinc RW 0x0							
gmacgrp_system_time_seconds 0x708	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	tss RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	tss RO 0x0															
gmacgrp_system_time_nanoseconds 0x70C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	tsss RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	tsss RO 0x0															

Register Address Offset	Bit Fields															
gmacgrp_system_time_seconds_update 0x710	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	tss RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	tss RW 0x0															
gmacgrp_system_time_nanoseconds_update 0x714	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addsub RW 0x0	tsss RW 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	tsss RW 0x0															
gmacgrp_timestamp_addend 0x718	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	tsar RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	tsar RW 0x0															
gmacgrp_target_time_seconds 0x71C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	tstr RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	tstr RW 0x0															
gmacgrp_target_time_nanoseconds 0x720	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	trgtbusy RO 0x0	ttslo RW 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ttslo RW 0x0															
gmacgrp_system_time_higher_word_seconds 0x724	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	tshwr RW 0x0															

Register Address Offset	Bit Fields															
gmacgrp_timestamp_status 0x728	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved		atsns RO 0x0					atsstm RO 0x0	Reserved				atsstn RO 0x0			
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved												tstrgterr RO 0x0	auxtstrig RO 0x0	tstargt RO 0x0	tssovf RO 0x0
gmacgrp_pps_control 0x72C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved									trgtmodsel0 RW 0x0		ppsen0 RW 0x0	ppsctrl_ppscmd RW 0x0			
gmacgrp_auxiliary_timestamp_nanoseconds 0x730	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	auxtslo RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	auxtslo RO 0x0															
gmacgrp_auxiliary_timestamp_seconds 0x734	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	auxtshi RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	auxtshi RO 0x0															
gmacgrp_pps0_interval 0x760	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ppsint RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ppsint RW 0x0															

Register Address Offset	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gmacgrp_pps0_width 0x764	ppswidth RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ppswidth RW 0x0															
gmacgrp_mac_address16_high 0x800	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address16_low 0x804	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address17_high 0x808	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address17_low 0x80C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address18_high 0x810	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address18_low 0x814	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address19_high 0x818	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address19_low 0x81C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address20_high 0x820	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address20_low 0x824	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address21_high 0x828	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address21_low 0x82C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address22_high 0x830	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address22_low 0x834	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address23_high 0x838	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address23_low 0x83C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address24_high 0x840	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address24_low 0x844	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address25_high 0x848	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address25_low 0x84C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address26_high 0x850	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address26_low 0x854	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address27_high 0x858	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address27_low 0x85C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address28_high 0x860	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address28_low 0x864	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address29_high 0x868	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address29_low 0x86C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address30_high 0x870	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address30_low 0x874	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address31_high 0x878	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address31_low 0x87C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address32_high 0x880	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address32_low 0x884	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address33_high 0x888	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address33_low 0x88C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address34_high 0x890	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address34_low 0x894	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address35_high 0x898	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address35_low 0x89C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address36_high 0x8A0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address36_low 0x8A4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address37_high 0x8A8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address37_low 0x8AC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address38_high 0x8B0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address38_low 0x8B4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address39_high 0x8B8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address39_low 0x8BC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address40_high 0x8C0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address40_low 0x8C4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address41_high 0x8C8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address41_low 0x8CC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address42_high 0x8D0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address42_low 0x8D4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address43_high 0x8D8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address43_low 0x8DC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address44_high 0x8E0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address44_low 0x8E4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address45_high 0x8E8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address45_low 0x8EC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address46_high 0x8F0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address46_low 0x8F4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address47_high 0x8F8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address47_low 0x8FC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address48_high 0x900	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address48_low 0x904	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address49_high 0x908	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address49_low 0x90C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address50_high 0x910	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address50_low 0x914	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address51_high 0x918	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address51_low 0x91C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address52_high 0x920	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address52_low 0x924	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address53_high 0x928	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address53_low 0x92C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address54_high 0x930	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address54_low 0x934	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address55_high 0x938	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address55_low 0x93C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address56_high 0x940	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address56_low 0x944	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address57_high 0x948	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address57_low 0x94C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address58_high 0x950	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address58_low 0x954	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address59_high 0x958	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address59_low 0x95C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address60_high 0x960	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address60_low 0x964	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address61_high 0x968	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address61_low 0x96C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address62_high 0x970	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address62_low 0x974	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address63_high 0x978	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address63_low 0x97C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address64_high 0x980	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address64_low 0x984	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address65_high 0x988	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address65_low 0x98C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address66_high 0x990	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address66_low 0x994	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address67_high 0x998	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address67_low 0x99C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gmacgrp_mac_address68_high 0x9A0	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address68_low 0x9A4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address69_high 0x9A8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address69_low 0x9AC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address70_high 0x9B0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address70_low 0x9B4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address71_high 0x9B8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address71_low 0x9BC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address72_high 0x9C0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address72_low 0x9C4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address73_high 0x9C8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address73_low 0x9CC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address74_high 0x9D0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address74_low 0x9D4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address75_high 0x9D8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address75_low 0x9DC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address76_high 0x9E0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address76_low 0x9E4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address77_high 0x9E8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address77_low 0x9EC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address78_high 0x9F0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address78_low 0x9F4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address79_high 0x9F8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address79_low 0x9FC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address80_high 0xA00	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register / Address Offset	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gmacgrp_mac_address80_low 0xA04	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address81_high 0xA08	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address81_low 0xA0C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address82_high 0xA10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address82_low 0xA14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address83_high 0xA18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address83_low 0xA1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address84_high 0xA20	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address84_low 0xA24	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address85_high 0xA28	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address85_low 0xA2C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address86_high 0xA30	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address86_low 0xA34	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address87_high 0xA38	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address87_low 0xA3C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gmacgrp_mac_address88_high 0xA40	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address88_low 0xA44	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address89_high 0xA48	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address89_low 0xA4C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address90_high 0xA50	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address90_low 0xA54	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address91_high 0xA58	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address91_low 0xA5C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address92_high 0xA60	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address92_low 0xA64	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gmacgrp_mac_address93_high 0xA68	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address93_low 0xA6C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address94_high 0xA70	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address94_low 0xA74	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address95_high 0xA78	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address95_low 0xA7C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address96_high 0xA80	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address96_low 0xA84	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address97_high 0xA88	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address97_low 0xA8C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gmacgrp_mac_address98_high 0xA90	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gmacgrp_mac_address98_low 0xA94	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gmacgrp_mac_address99_high 0xA98	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gmacgrp_mac_address99_low 0xA9C	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gmacgrp_mac_address100_high 0xAA0	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address100_low 0xAA4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address101_high 0xAA8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address101_low 0xAAC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address102_high 0xAB0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address102_low 0xAB4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address103_high 0xAB8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address103_low 0xABC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address104_high 0xAC0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address104_low 0xAC4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address105_high 0xAC8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address105_low 0xACC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address106_high 0xAD0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address106_low 0xAD4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address107_high 0xAD8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address107_low 0xADC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address108_high 0xAE0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address108_low 0xAE4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address109_high 0xAE8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address109_low 0xAEC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address110_high 0xAF0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gmacgrp_mac_address110_low 0xAF4	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address111_high 0xAF8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address111_low 0xAFC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address112_high 0xB00	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address112_low 0xB04	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address113_high 0xB08	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address113_low 0xB0C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address114_high 0xB10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address114_low 0xB14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address115_high 0xB18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address115_low 0xB1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address116_high 0xB20	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address116_low 0xB24	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address117_high 0xB28	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address117_low 0xB2C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address118_high 0xB30	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address118_low 0xB34	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address119_high 0xB38	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address119_low 0xB3C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address120_high 0xB40	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address120_low 0xB44	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address121_high 0xB48	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address121_low 0xB4C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address122_high 0xB50	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address122_low 0xB54	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address123_high 0xB58	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address123_low 0xB5C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address124_high 0xB60	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address124_low 0xB64	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address125_high 0xB68	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address125_low 0xB6C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address126_high 0xB70	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address126_low 0xB74	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address127_high 0xB78	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address127_low 0xB7C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
dmagrp_bus_mode 0x1000	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	rib RW 0x0	reserved_30 RO 0x0	prwg RW 0x0		txpr RW 0x0	mb RW 0x0	aal RW 0x0	eightxpbl RW 0x0	usp RW 0x0	rpbl RW 0x1						fb RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	pr RW 0x0		pbl RW 0x1						atds RW 0x0	dsl RW 0x0					da RW 0x0	swr RW 0x1
dmagrp_transmit_poll_demand 0x1004	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	tpd RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	tpd RW 0x0															
dmagrp_receive_poll_demand 0x1008	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	rpd RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rpd RW 0x0															
dmagrp_receive_descriptor_list_address 0x100C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	rdesla_32bit RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rdesla_32bit RW 0x0														Reserved	
dmagrp_transmit_descriptor_list_address 0x1010	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	tdesla_32bit RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	tdesla_32bit RW 0x0														Reserved	

Register Address Offset	Bit Fields															
dmagrp_status 0x1014	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31 RO 0x0	glpii RO 0x0	tti RO 0x0	gpi RO 0x0	gmi RO 0x0	gli RO 0x0	eb RO 0x0			ts RO 0x0			rs RO 0x0			nis RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ais RW 0x0	eri RW 0x0	fbi RW 0x0	reserved_12_11 RO 0x0		eti RW 0x0	rwt RW 0x0	rps RW 0x0	ru RW 0x0	ri RW 0x0	unf RW 0x0	ovf RW 0x0	tjt RW 0x0	tu RW 0x0	tps RW 0x0	ti RW 0x0
dmagrp_operation_mode 0x1018	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_27 RO 0x0					dt RW 0x0	rsf RW 0x0	dff RW 0x0	rfa_2 RO 0x0	rfd_2 RO 0x0	tsf RW 0x0	ftf RW 0x0	reserved_19_17 RO 0x0			ttc RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ttc RW 0x0		st RW 0x0	rfd RO 0x0		rfa RO 0x0		efc RO 0x0	fef RW 0x0	fuf RW 0x0	dgf RW 0x0	rtc RW 0x0		osf RW 0x0	sr RW 0x0	reserved_0 RO 0x0
dmagrp_interrupt_enable 0x101C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_17 RO 0x0															nie RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	aie RW 0x0	ere RW 0x0	fbe RW 0x0	reserved_12_11 RO 0x0		ete RW 0x0	rwe RW 0x0	rse RW 0x0	rue RW 0x0	rie RW 0x0	une RW 0x0	ove RW 0x0	tje RW 0x0	tue RW 0x0	tse RW 0x0	tie RW 0x0
dmagrp_missed_frame_and_buffer_overflow_counter 0x1020	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_29 RO 0x0			ovfcntovf RO 0x0	ovffrmcnt RO 0x0											miscntovf RO 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	misfrmcnt RO 0x0															
dmagrp_receive_interrupt_watchdog_timer 0x1024	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_8 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved_31_8 RO 0x0								riwt RW 0x0							

Register Address Offset	Bit Fields															
dmagrp_axi_bus_mode 0x1028	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	en_lpi RW 0x0	lpi_xit_frm RW 0x0	Reserved						wr_osr_lmt RW 0x1				rd_osr_lmt RW 0x1			
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		onekbbe RW 0x0	axi_aal RO 0x0	Reserved								blen16 RW 0x0	blen8 RW 0x0	blen4 RW 0x0	undefined RO 0x1
dmagrp_ahb_or_axi_status 0x102C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_2 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved_31_2 RO 0x0														axirdsts RO 0x0	axwhsts RO 0x0
dmagrp_current_host_transmit_descriptor 0x1048	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	curtdesaptr RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	curtdesaptr RO 0x0															
dmagrp_current_host_receive_descriptor 0x104C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	currdesaptr RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	currdesaptr RO 0x0															
dmagrp_current_host_transmit_buffer_address 0x1050	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	curtbufaptr RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	curtbufaptr RO 0x0															

Register Address Offset	Bit Fields															
dmagrp_current_host_receive_buffer_address 0x1054	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	currbufaptr RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	currbufaptr RO 0x0															
dmagrp_hw_feature 0x1058	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31 RO 0x0	actphyif RO 0x0			savlanins RO 0x1	flexippsen RO 0x0	inttsen RO 0x0	enhdessel RO 0x1	txchcnt RO 0x1		rxchcnt RO 0x0		rxfifosize RO 0x0	rxtyp2coe RO 0x1	rxtyp1coe RO 0x0	txoesel RO 0x1
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	avsel RO 0x1	eeesel RO 0x1	tsver2sel RO 0x1	tsver1sel RO 0x0	mmcsel RO 0x1	mgksel RO 0x1	rwksel RO 0x0	smasel RO 0x1	l3l4fltren RO 0x1	pcssel RO 0x0	addmacadrsel RO 0x1	hashsel RO 0x0	exthashen RO 0x0	hdsel RO 0x1	gmiisel RO 0x1	miisel RO 0x1
i_emac_emac1																
gmacgrp_mac_configuration 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31 RO 0x0	sarc RW 0x0			twokpe RW 0x0	sfterr RO 0x0	cst RW 0x0	tc RO 0x0	wd RW 0x0	jd RW 0x0	be RW 0x0	je RW 0x0	ifg RW 0x0			dcrs RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ps RW 0x0	fes RW 0x0	do RW 0x0	lm RW 0x0	dm RW 0x0	ipc RW 0x0	dr RW 0x0	lud RO 0x0	acs RW 0x0	bl RW 0x0		dc RW 0x0	te RW 0x0	re RW 0x0	prelen RW 0x0	
gmacgrp_mac_frame_filter 0x4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ra RW 0x0	reserved_30_22 RO 0x0									dntu RW 0x0	ipfe RW 0x0	reserved_19_17 RO 0x0			vtfe RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved_15_11 RO 0x0					hpf RO 0x0	saf RW 0x0	saif RW 0x0	pcf RW 0x0		dbf RW 0x0	pm RW 0x0	daif RW 0x0	hmc RO 0x0	huc RO 0x0	pr RW 0x0

Register Address Offset	Bit Fields															
gmacgrp_gmii_address 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_16 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	pa RW 0x0					gr RW 0x0					cr RW 0x0				gw RW 0x0	gb RW 0x0
gmacgrp_gmii_data 0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_16 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	gd RW 0x0															
gmacgrp_flow_control 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	pt RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved_15_8 RO 0x0								dzpq RW 0x0	reserved_6 RO 0x0	plt RW 0x0		up RW 0x0	rfe RW 0x0	tfe RW 0x0	fca_bpa RW 0x0
gmacgrp_vlan_tag 0x1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_20 RO 0x0												vthm RO 0x0	esvl RW 0x0	vtim RW 0x0	etv RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	vl RW 0x0															
gmacgrp_version 0x20	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_16 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	userver RO 0x10								snpsver RO 0x37							

Register Address Offset	Bit Fields															
gmacgrp_debug 0x24	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_26 RO 0x0						txstsfsts RO 0x0	txfsts RO 0x0	reserved_23 RO 0x0	twcsts RO 0x0	trcsts RO 0x0		txpaused RO 0x0	tfcsts RO 0x0		tpests RO 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved_15_10 RO 0x0						rxfsts RO 0x0		reserved_7 RO 0x0	rrcsts RO 0x0		rwcsts RO 0x0	reserved_3 RO 0x0	rfcfcsts RO 0x0		rpests RO 0x0
gmacgrp_lpi_control_status 0x30	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_20 RO 0x0												lpitxa RW 0x0	plsen RO 0x0	pls RW 0x0	lpien RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved_15_10 RO 0x0						rlpist RO 0x0	tlpist RO 0x0	reserved_7_4 RO 0x0				rlpiex RO 0x0	rlpien RO 0x0	tlpiex RO 0x0	tlpien RO 0x0
gmacgrp_lpi_timers_control 0x34	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_26 RO 0x0						lst RW 0x3E8									
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	twt RW 0x0															
gmacgrp_interrupt_status 0x38	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_12 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved_31_12 RO 0x0				gpiis RO 0x0	lpiis RO 0x0	tsis RO 0x0	reserved_8 RO 0x0	mmcrxipis RO 0x0	mmctxis RO 0x0	mmcrxis RO 0x0	mmcis RO 0x0	pmtis RO 0x0	pcsancis RO 0x0	pcslchgis RO 0x0	rgsmiiis RO 0x0

Register Address Offset	Bit Fields															
gmacgrp_interrupt_mask 0x3C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_11 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved_31_11 RO 0x0					lpiim RW 0x0	tsim RO 0x0	reserved_8_4 RO 0x0					pmtim RW 0x0	pcsancim RO 0x0	pcslchgim RO 0x0	rgsmiiim RO 0x0
gmacgrp_mac_address0_high 0x40	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RO 0x1	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address0_low 0x44	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address1_high 0x48	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address1_low 0x4C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address2_high 0x50	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address2_low 0x54	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address3_high 0x58	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address3_low 0x5C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address4_high 0x60	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address4_low 0x64	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address5_high 0x68	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address5_low 0x6C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address6_high 0x70	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address6_low 0x74	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address7_high 0x78	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address7_low 0x7C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address8_high 0x80	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address8_low 0x84	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address9_high 0x88	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address9_low 0x8C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address10_high 0x90	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address10_low 0x94	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address11_high 0x98	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address11_low 0x9C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address12_high 0xA0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address12_low 0xA4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address13_high 0xA8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address13_low 0xAC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address14_high 0xB0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address14_low 0xB4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address15_high 0xB8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address15_low 0xBC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_sgmii_rgmii_smii_control_status 0xD8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved												lnksts RO 0x0	lnkspeed RO 0x0		lnkmod RO 0x0
gmacgrp_wdog_timeout 0xDC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_17 RO 0x0															pwe RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved_15_14 RO 0x0		wto RW 0x0													

Register Address Offset	Bit Fields															
gmacgrp_gen pio 0xE0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_x RO 0x0							gpit RW 0x0	reserved_23_x RO 0x0							gpie RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved_15_x RO 0x0							gpo RW 0x0	reserved_7_x RO 0x0							gpis RO 0x0
gmacgrp_mmc _control 0x100	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_9 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved_31_9 RO 0x0							ucdbc RW 0x0	reserved_7_6 RO 0x0		cntprstlvl RW 0x0	cntprst RW 0x0	cntfreez RW 0x0	rstonrd RW 0x0	cntstopro RW 0x0	cntrst RW 0x0
gmacgrp_mmc _receive_in terrupt 0x104	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_26 RO 0x0						rxctrlfis RO 0x0	rxrcverrfis RO 0x0	rxwdogfis RO 0x0	rxvlangbfis RO 0x0	rxfovfis RO 0x0	rxpausfis RO 0x0	rxorangefis RO 0x0	rxlenerfis RO 0x0	rxucgfis RO 0x0	rx1024tmaxoctgbfis RO 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rx512t1023octgbfis RO 0x0	rx256t511octgbfis RO 0x0	rx128t255octgbfis RO 0x0	rx65t127octgbfis RO 0x0	rx64octgbfis RO 0x0	rxosizegfis RO 0x0	rxusizegfis RO 0x0	rxjaberfis RO 0x0	rxruntfis RO 0x0	rxalignerfis RO 0x0	rxcrcerfis RO 0x0	rxmcgfis RO 0x0	rxbcgfis RO 0x0	rxgoctis RO 0x0	rxgboctis RO 0x0	rxgbfrmis RO 0x0
gmacgrp_mmc _transmit_i nterrupt 0x108	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_26 RO 0x0						txosizegfis RO 0x0	txvlangfis RO 0x0	txpausfis RO 0x0	txexdeffis RO 0x0	txgfrmis RO 0x0	txgoctis RO 0x0	txcarerfis RO 0x0	txexcolfis RO 0x0	txlatcolfis RO 0x0	txdeffis RO 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	txmcolgfis RO 0x0	txscolgfis RO 0x0	txuflowerfis RO 0x0	txbcgbfis RO 0x0	txmcgbfis RO 0x0	txucgbfis RO 0x0	tx1024tmaxoctgbfis RO 0x0	tx512t1023octgbfis RO 0x0	tx256t511octgbfis RO 0x0	tx128t255octgbfis RO 0x0	tx65t127octgbfis RO 0x0	tx64octgbfis RO 0x0	txmcgfis RO 0x0	txbcgfis RO 0x0	txgbfrmis RO 0x0	txgboctis RO 0x0

Register Address Offset	Bit Fields															
gmacgrp_mmc_receive_interrupt_mask 0x10C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_26 RO 0x0						rxctrlfim RW 0x0	rxrcverrfim RW 0x0	rxwdogfim RW 0x0	rxvlangbfim RW 0x0	rxfovfim RW 0x0	rxpausfim RW 0x0	rxorangefim RW 0x0	rxlenerfim RW 0x0	rxucgfim RW 0x0	rx1024tmaxoctgbfim RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rx512t1023octgbfim RW 0x0	rx256t511octgbfim RW 0x0	rx128t255octgbfim RW 0x0	rx65t127octgbfim RW 0x0	rx64octgbfim RW 0x0	rxosizegfim RW 0x0	rxusizegfim RW 0x0	rxjaberfim RW 0x0	rxruntfim RW 0x0	rxalignerfim RW 0x0	rxcrcerfim RW 0x0	rxmcgfim RW 0x0	rxbcgfim RW 0x0	rxgoctim RW 0x0	rxgboctim RW 0x0	rxgbfrmim RW 0x0
gmacgrp_mmc_transmit_interrupt_mask 0x110	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_26 RO 0x0						txosizegfim RW 0x0	txvlangfim RW 0x0	txpausfim RW 0x0	txexdeffim RW 0x0	txgfrmim RW 0x0	txgoctim RW 0x0	txcarerfim RW 0x0	txexcolfim RW 0x0	txlatcolfim RW 0x0	txdeffim RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	txmcolgfim RW 0x0	txscolgfim RW 0x0	txuflowerfim RW 0x0	txbcgbfim RW 0x0	txmcgbfim RW 0x0	txucgbfim RW 0x0	tx1024tmaxoctgbfim RW 0x0	tx512t1023octgbfim RW 0x0	tx256t511octgbfim RW 0x0	tx128t255octgbfim RW 0x0	tx65t127octgbfim RW 0x0	tx64octgbfim RW 0x0	txmcgfim RW 0x0	txbcgfim RW 0x0	txgbfrmim RW 0x0	txgboctim RW 0x0
gmacgrp_txoctetcount_gb 0x114	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_txframecount_gb 0x118	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															

Register Address Offset	Bit Fields															
gmacgrp_txbroadcast-frames_g 0x11C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_txmulticast-frames_g 0x120	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_tx64octets_gb 0x124	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_tx65to127octets_gb 0x128	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_tx128to255octets_gb 0x12C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_tx256to511octets_gb 0x130	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															

Register Address Offset	Bit Fields															
gmacgrp_tx512to1023octets_gb 0x134	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_tx1024tomaxoctets_gb 0x138	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_txunicastframes_gb 0x13C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_txmulticast-frames_gb 0x140	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_txbroadcast-frames_gb 0x144	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_txunderflo-werror 0x148	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															

Register Address Offset	Bit Fields															
gmacgrp_txsinglecol_g 0x14C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_txmulticol_g 0x150	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_txdeferred 0x154	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_txlatecol 0x158	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_txexesscol 0x15C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_txcarriererr 0x160	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															

Register Address Offset	Bit Fields															
gmacgrp_txoctetcnt 0x164	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	txoctetcount_g RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	txoctetcount_g RO 0x0															
gmacgrp_txframecount_g 0x168	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_txexcessdef 0x16C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_txpauseframes 0x170	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_txvlanframes_g 0x174	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_txoversize_g 0x178	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															

Register Address Offset	Bit Fields															
gmacgrp_rxframecount_gb 0x180	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxoctetcount_gb 0x184	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxoctetcount_g 0x188	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxbroadcast-frames_g 0x18C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxmulticast-frames_g 0x190	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxcrcerror 0x194	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															

Register Address Offset	Bit Fields															
gmacgrp_rxalignmenterror 0x198	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxrunterror 0x19C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxjabbererror 0x1A0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxundersize_g 0x1A4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxoversize_g 0x1A8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rx64octets_gb 0x1AC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															

Register Address Offset	Bit Fields															
gmacgrp_rx65to127octets_gb 0x1B0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rx128to255octets_gb 0x1B4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rx256to511octets_gb 0x1B8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rx512to1023octets_gb 0x1BC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rx1024tomaxoctets_gb 0x1C0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxunicastframes_g 0x1C4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															

Register Address Offset	Bit Fields															
gmacgrp_rxlengtherror 0x1C8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxoutofrange-type 0x1CC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxpauseframes 0x1D0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxfifooverflow 0x1D4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxvlanframes_gb 0x1D8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxwatchdogerror 0x1DC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															

Register Address Offset	Bit Fields															
gmacgrp_rxrcverror 0x1E0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxctrlframes_g 0x1E4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_mmc_ipc_receive_interrupt_mask 0x200	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved		rxicmperoim RW 0x0	rxicmpgoim RW 0x0	rxtcperoim RW 0x0	rxtcpgoim RW 0x0	rxudperoim RW 0x0	rxudpgoim RW 0x0	rxipv6nopayoim RW 0x0	rxipv6heroim RW 0x0	rxipv6goim RW 0x0	rxipv4udsbloim RW 0x0	rxipv4fragoim RW 0x0	rxipv4nopayoim RW 0x0	rxipv4heroim RW 0x0	rxipv4goim RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		rxicmperfim RW 0x0	rxicmpgfim RW 0x0	rxtcperfim RW 0x0	rxtcpgfim RW 0x0	rxudperfim RW 0x0	rxudpgfim RW 0x0	rxipv6nopayfim RW 0x0	rxipv6herfim RW 0x0	rxipv6gfim RW 0x0	rxipv4udsblfim RW 0x0	rxipv4fragfim RW 0x0	rxipv4nopayfim RW 0x0	rxipv4herfim RW 0x0	rxipv4gfim RW 0x0
gmacgrp_mmc_ipc_receive_interrupt 0x208	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved		rxicmperois RO 0x0	rxicmpgois RO 0x0	rxtcperois RO 0x0	rxtcpgois RO 0x0	rxudperois RO 0x0	rxudpgois RO 0x0	rxipv6nopayois RO 0x0	rxipv6herois RO 0x0	rxipv6gois RO 0x0	rxipv4udsbloois RO 0x0	rxipv4fragois RO 0x0	rxipv4nopayois RO 0x0	rxipv4herois RO 0x0	rxipv4gois RO 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		rxicmperfis RO 0x0	rxicmpgfis RO 0x0	rxtcperfis RO 0x0	rxtcpgfis RO 0x0	rxudperfis RO 0x0	rxudpgfis RO 0x0	rxipv6nopayfis RO 0x0	rxipv6herfis RO 0x0	rxipv6gfis RO 0x0	rxipv4udsblfis RO 0x0	rxipv4fragfis RO 0x0	rxipv4nopayfis RO 0x0	rxipv4herfis RO 0x0	rxipv4gfis RO 0x0

Register Address Offset	Bit Fields															
gmacgrp_rxipv4_gd_frms 0x210	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxipv4_hdrerr_frms 0x214	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxipv4_nopay_frms 0x218	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxipv4_frag_frms 0x21C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxipv4_udsbl_frms 0x220	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxipv6_gd_frms 0x224	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															

Register Address Offset	Bit Fields															
gmacgrp_rxipv6_hdrerr_frms 0x228	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxipv6_nopay_frms 0x22C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxudp_gd_frms 0x230	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxudp_err_frms 0x234	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxtcp_gd_frms 0x238	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxtcp_err_frms 0x23C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															

Register Address Offset	Bit Fields															
gmacgrp_rxicmp_gd_frms 0x240	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxicmp_err_frms 0x244	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxipv4_gd_octets 0x250	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxipv4_hdrerr_octets 0x254	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxipv4_nopay_octets 0x258	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxipv4_frag_octets 0x25C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															

Register Address Offset	Bit Fields															
gmacgrp_rxipv4_udsbl_octets 0x260	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxipv6_gd_octets 0x264	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxipv6_hdrerr_octets 0x268	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxipv6_nopay_octets 0x26C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxudp_gd_octets 0x270	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxudp_err_octets 0x274	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															

Register Address Offset	Bit Fields															
gmacgrp_rxtcp_gd_octets 0x278	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxtcperroctets 0x27C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	rxtcp_err_octets RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rxtcp_err_octets RO 0x0															
gmacgrp_rxicmp_gd_octets 0x280	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxicmp_err_octets 0x284	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_l3_l4_control0 0x400	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_22 RO 0x0										l4dpim0 RW 0x0	l4dpm0 RW 0x0	l4spim0 RW 0x0	l4spm0 RW 0x0	reserved_17 RO 0x0	l4pen0 RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l3hdbm0 RW 0x0					l3hsbm0 RW 0x0					l3daim0 RW 0x0	l3dam0 RW 0x0	l3saim0 RW 0x0	l3sam0 RW 0x0	reserved_1 RO 0x0	l3pen0 RW 0x0

Register Address Offset	Bit Fields															
gmacgrp_layer4_address0 0x404	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	l4dp0 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l4sp0 RW 0x0															
gmacgrp_layer3_addr0_reg0 0x410	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	l3a00 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l3a00 RW 0x0															
gmacgrp_layer3_addr1_reg0 0x414	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	l3a10 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l3a10 RW 0x0															
gmacgrp_layer3_addr2_reg0 0x418	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	l3a20 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l3a20 RW 0x0															
gmacgrp_layer3_addr3_reg0 0x41C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	l3a30 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l3a30 RW 0x0															

Register Address Offset	Bit Fields															
gmacgrp_l3_l4_control1 0x430	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved										l4dpim1 RW 0x0	l4dpm1 RW 0x0	l4spim1 RW 0x0	l4spm1 RW 0x0	Reserved	l4pen1 RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l3hdbm1 RW 0x0					l3hsbm1 RW 0x0					l3daim1 RW 0x0	l3dam1 RW 0x0	l3saim1 RW 0x0	l3sam1 RW 0x0	Reserved	l3pen1 RW 0x0
gmacgrp_layer4_address1 0x434	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	l4dp1 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l4sp1 RW 0x0															
gmacgrp_layer3_addr0_reg1 0x440	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	l3a01 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l3a01 RW 0x0															
gmacgrp_layer3_addr1_reg1 0x444	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	l3a11 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l3a11 RW 0x0															
gmacgrp_layer3_addr2_reg1 0x448	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	l3a21 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l3a21 RW 0x0															

Register Address Offset	Bit Fields															
gmacgrp_layer3_addr3_reg1 0x44C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	l3a31 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l3a31 RW 0x0															
gmacgrp_l3_l4_control2 0x460	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved										l4dpim2 RW 0x0	l4dpm2 RW 0x0	l4spim2 RW 0x0	l4spm2 RW 0x0	Reserved	l4pen2 RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l3hdbm2 RW 0x0					l3hsbm2 RW 0x0					l3daim2 RW 0x0	l3dam2 RW 0x0	l3saim2 RW 0x0	l3sam2 RW 0x0	Reserved	l3pen2 RW 0x0
gmacgrp_layer4_address2 0x464	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	l4dp2 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l4sp2 RW 0x0															
gmacgrp_layer3_addr0_reg2 0x470	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	l3a02 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l3a02 RW 0x0															
gmacgrp_layer3_addr1_reg2 0x474	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	l3a12 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l3a12 RW 0x0															

Register Address Offset	Bit Fields															
gmacgrp_layer3_addr2_reg2 0x478	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	l3a22 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l3a22 RW 0x0															
gmacgrp_layer3_addr3_reg2 0x47C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	l3a32 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l3a32 RW 0x0															
gmacgrp_l3_l4_control3 0x490	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved										l4dpim3 RW 0x0	l4dpm3 RW 0x0	l4spim3 RW 0x0	l4spm3 RW 0x0	Reserved	l4pen3 RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l3hdbm3 RW 0x0					l3hsbm3 RW 0x0					l3daim3 RW 0x0	l3dam3 RW 0x0	l3saim3 RW 0x0	l3sam3 RW 0x0	Reserved	l3pen3 RW 0x0
gmacgrp_layer4_address3 0x494	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	l4dp3 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l4sp3 RW 0x0															
gmacgrp_layer3_addr0_reg3 0x4A0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	l3a03 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l3a03 RW 0x0															

Register Address Offset	Bit Fields															
gmacgrp_layer3_addr1_reg3 0x4A4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	l3a13 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l3a13 RW 0x0															
gmacgrp_layer3_addr2_reg3 0x4A8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	l3a23 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l3a23 RW 0x0															
gmacgrp_layer3_addr3_reg3 0x4AC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	l3a33 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l3a33 RW 0x0															
gmacgrp_hash_table_reg0 0x500	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ht31t0 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ht31t0 RW 0x0															
gmacgrp_hash_table_reg1 0x504	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ht63t32 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ht63t32 RW 0x0															
gmacgrp_hash_table_reg2 0x508	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ht95t64 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ht95t64 RW 0x0															

Register Address Offset	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gmacgrp_hash_table_reg3 0x50C	ht127t96 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ht127t96 RW 0x0															
gmacgrp_hash_table_reg4 0x510	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ht159t128 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ht159t128 RW 0x0															
gmacgrp_hash_table_reg5 0x514	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ht191t160 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ht191t160 RW 0x0															
gmacgrp_hash_table_reg6 0x518	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ht223t196 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ht223t196 RW 0x0															
gmacgrp_hash_table_reg7 0x51C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ht255t224 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ht255t224 RW 0x0															
gmacgrp_vlan_incl_reg 0x584	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_20 RO 0x0												csvl RW 0x0	vlp RW 0x0	vlc RW 0x0	
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	vlt RW 0x0															

Register Address Offset	Bit Fields															
gmacgrp_vlan_hash_table_reg 0x588	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	vlht RW 0x0															
gmacgrp_timestamp_control 0x700	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_29 RO 0x0			atsen3 RO 0x0	atsen2 RO 0x0	atsen1 RO 0x0	atsen0 RO 0x0	atsfc RO 0x0	reserved_23_19 RO 0x0					tsenmacaddr RW 0x0	snaptypsel RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	tsmstrena RW 0x0	tsevntena RW 0x0	tsipv4ena RW 0x1	tsipv6ena RW 0x0	tsipena RW 0x0	tsver2ena RW 0x0	tsctrlssr RW 0x0	tsenall RW 0x0	reserved_7_6 RO 0x0		tsaddreg RO 0x0	tstrig RO 0x0	tsupdt RO 0x0	tsinit RO 0x0	tscfupdt RO 0x0	tsena RW 0x0
gmacgrp_sub_second_increment 0x704	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								ssinc RW 0x0							
gmacgrp_system_time_seconds 0x708	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	tss RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	tss RO 0x0															
gmacgrp_system_time_nanoseconds 0x70C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	tsss RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	tsss RO 0x0															

Register Address Offset	Bit Fields															
gmacgrp_system_time_seconds_update 0x710	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	tss RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	tss RW 0x0															
gmacgrp_system_time_nanoseconds_update 0x714	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addsub RW 0x0	tsss RW 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	tsss RW 0x0															
gmacgrp_timestamp_addend 0x718	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	tsar RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	tsar RW 0x0															
gmacgrp_target_time_seconds 0x71C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	tstr RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	tstr RW 0x0															
gmacgrp_target_time_nanoseconds 0x720	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	trgtbusy RO 0x0	ttslo RW 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ttslo RW 0x0															
gmacgrp_system_time_higher_word_seconds 0x724	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	tshwr RW 0x0															

Register Address Offset	Bit Fields															
gmacgrp_timestamp_status 0x728	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved			atsns RO 0x0				atsstm RO 0x0	Reserved				atsstn RO 0x0			
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved												tstrgterr RO 0x0	auxtstrig RO 0x0	tstargt RO 0x0	tssovf RO 0x0
gmacgrp_pps_control 0x72C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved									trgtmodsel0 RW 0x0		ppsen0 RW 0x0	ppsctrl_ppscmd RW 0x0			
gmacgrp_auxiliary_timestamp_nanoseconds 0x730	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	auxtslo RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	auxtslo RO 0x0															
gmacgrp_auxiliary_timestamp_seconds 0x734	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	auxtshi RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	auxtshi RO 0x0															
gmacgrp_pps0_interval 0x760	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ppsint RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ppsint RW 0x0															

Register Address Offset	Bit Fields															
gmacgrp_pps0_width 0x764	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ppswidth RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ppswidth RW 0x0															
gmacgrp_mac_address16_high 0x800	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address16_low 0x804	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address17_high 0x808	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address17_low 0x80C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address18_high 0x810	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address18_low 0x814	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address19_high 0x818	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address19_low 0x81C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address20_high 0x820	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address20_low 0x824	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address21_high 0x828	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address21_low 0x82C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address22_high 0x830	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address22_low 0x834	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address23_high 0x838	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address23_low 0x83C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address24_high 0x840	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address24_low 0x844	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address25_high 0x848	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address25_low 0x84C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address26_high 0x850	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address26_low 0x854	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address27_high 0x858	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address27_low 0x85C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address28_high 0x860	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address28_low 0x864	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address29_high 0x868	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address29_low 0x86C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address30_high 0x870	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address30_low 0x874	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address31_high 0x878	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address31_low 0x87C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address32_high 0x880	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address32_low 0x884	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address33_high 0x888	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address33_low 0x88C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address34_high 0x890	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address34_low 0x894	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address35_high 0x898	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address35_low 0x89C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address36_high 0x8A0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address36_low 0x8A4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address37_high 0x8A8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address37_low 0x8AC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address38_high 0x8B0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address38_low 0x8B4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address39_high 0x8B8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address39_low 0x8BC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address40_high 0x8C0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address40_low 0x8C4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address41_high 0x8C8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address41_low 0x8CC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address42_high 0x8D0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address42_low 0x8D4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gmacgrp_mac_address43_high 0x8D8	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address43_low 0x8DC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address44_high 0x8E0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address44_low 0x8E4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address45_high 0x8E8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address45_low 0x8EC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address46_high 0x8F0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address46_low 0x8F4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address47_high 0x8F8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address47_low 0x8FC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address48_high 0x900	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address48_low 0x904	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address49_high 0x908	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address49_low 0x90C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address50_high 0x910	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address50_low 0x914	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address51_high 0x918	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address51_low 0x91C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address52_high 0x920	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address52_low 0x924	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address53_high 0x928	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address53_low 0x92C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address54_high 0x930	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address54_low 0x934	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address55_high 0x938	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address55_low 0x93C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address56_high 0x940	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address56_low 0x944	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address57_high 0x948	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address57_low 0x94C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gmacgrp_mac_address58_high 0x950	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address58_low 0x954	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address59_high 0x958	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address59_low 0x95C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address60_high 0x960	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address60_low 0x964	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address61_high 0x968	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address61_low 0x96C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address62_high 0x970	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address62_low 0x974	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address63_high 0x978	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address63_low 0x97C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address64_high 0x980	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address64_low 0x984	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address65_high 0x988	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address65_low 0x98C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address66_high 0x990	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address66_low 0x994	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address67_high 0x998	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address67_low 0x99C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address68_high 0x9A0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address68_low 0x9A4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address69_high 0x9A8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address69_low 0x9AC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address70_high 0x9B0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address70_low 0x9B4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address71_high 0x9B8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address71_low 0x9BC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address72_high 0x9C0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address72_low 0x9C4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address73_high 0x9C8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address73_low 0x9CC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address74_high 0x9D0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address74_low 0x9D4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address75_high 0x9D8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address75_low 0x9DC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address76_high 0x9E0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address76_low 0x9E4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address77_high 0x9E8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address77_low 0x9EC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address78_high 0x9F0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address78_low 0x9F4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address79_high 0x9F8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address79_low 0x9FC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address80_high 0xA00	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address80_low 0xA04	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address81_high 0xA08	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address81_low 0xA0C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address82_high 0xA10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address82_low 0xA14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gmacgrp_mac_address83_high 0xA18	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gmacgrp_mac_address83_low 0xA1C	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gmacgrp_mac_address84_high 0xA20	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gmacgrp_mac_address84_low 0xA24	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gmacgrp_mac_address85_high 0xA28	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address85_low 0xA2C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address86_high 0xA30	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address86_low 0xA34	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address87_high 0xA38	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address87_low 0xA3C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address88_high 0xA40	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address88_low 0xA44	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address89_high 0xA48	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address89_low 0xA4C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address90_high 0xA50	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address90_low 0xA54	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address91_high 0xA58	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address91_low 0xA5C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address92_high 0xA60	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address92_low 0xA64	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address93_high 0xA68	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address93_low 0xA6C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address94_high 0xA70	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address94_low 0xA74	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address95_high 0xA78	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address95_low 0xA7C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address96_high 0xA80	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address96_low 0xA84	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address97_high 0xA88	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address97_low 0xA8C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gmacgrp_mac_address98_high 0xA90	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gmacgrp_mac_address98_low 0xA94	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gmacgrp_mac_address99_high 0xA98	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gmacgrp_mac_address99_low 0xA9C	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gmacgrp_mac_address100_high 0xAA0	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address100_low 0xAA4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address101_high 0xAA8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address101_low 0xAAC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address102_high 0xAB0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address102_low 0xAB4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address103_high 0xAB8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address103_low 0xABC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address104_high 0xAC0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address104_low 0xAC4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address105_high 0xAC8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address105_low 0xACC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address106_high 0xAD0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address106_low 0xAD4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address107_high 0xAD8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address107_low 0xADC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address108_high 0xAE0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address108_low 0xAE4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address109_high 0xAE8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address109_low 0xAEC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address110_high 0xAF0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address110_low 0xAF4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address111_high 0xAF8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address111_low 0xAFC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address112_high 0xB00	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address112_low 0xB04	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address113_high 0xB08	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address113_low 0xB0C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address114_high 0xB10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address114_low 0xB14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address115_high 0xB18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address115_low 0xB1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address116_high 0xB20	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address116_low 0xB24	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address117_high 0xB28	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address117_low 0xB2C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address118_high 0xB30	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address118_low 0xB34	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address119_high 0xB38	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address119_low 0xB3C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address120_high 0xB40	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address120_low 0xB44	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address121_high 0xB48	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address121_low 0xB4C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address122_high 0xB50	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address122_low 0xB54	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address123_high 0xB58	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address123_low 0xB5C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address124_high 0xB60	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address124_low 0xB64	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address125_high 0xB68	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address125_low 0xB6C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address126_high 0xB70	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address126_low 0xB74	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address127_high 0xB78	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address127_low 0xB7C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
dmagrp_bus_mode 0x1000	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	rib RW 0x0	reserved_30 RO 0x0	prwg RW 0x0		txpr RW 0x0	mb RW 0x0	aal RW 0x0	eighxtxpbl RW 0x0	usp RW 0x0	rpbl RW 0x1						fb RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	pr RW 0x0		pbl RW 0x1						atds RW 0x0	dsl RW 0x0					da RW 0x0	swr RW 0x1
dmagrp_transmit_poll_demand 0x1004	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	tpd RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	tpd RW 0x0															
dmagrp_receive_poll_demand 0x1008	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	rpd RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rpd RW 0x0															
dmagrp_receive_descriptor_list_address 0x100C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	rdesla_32bit RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rdesla_32bit RW 0x0														Reserved	
dmagrp_transmit_descriptor_list_address 0x1010	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	tdesla_32bit RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	tdesla_32bit RW 0x0														Reserved	

Register Address Offset	Bit Fields															
dmagrp_status 0x1014	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31 RO 0x0	glpii RO 0x0	tti RO 0x0	gpi RO 0x0	gmi RO 0x0	gli RO 0x0	eb RO 0x0			ts RO 0x0			rs RO 0x0			nis RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ais RW 0x0	eri RW 0x0	fbi RW 0x0	reserved_12_11 RO 0x0		eti RW 0x0	rwt RW 0x0	rps RW 0x0	ru RW 0x0	ri RW 0x0	unf RW 0x0	ovf RW 0x0	tjt RW 0x0	tu RW 0x0	tps RW 0x0	ti RW 0x0
dmagrp_operation_mode 0x1018	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_27 RO 0x0					dt RW 0x0	rsf RW 0x0	dff RW 0x0	rfa_2 RO 0x0	rfd_2 RO 0x0	tsf RW 0x0	ftf RW 0x0	reserved_19_17 RO 0x0			ttc RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ttc RW 0x0		st RW 0x0	rfd RO 0x0		rfa RO 0x0		efc RO 0x0	fef RW 0x0	fuf RW 0x0	dgf RW 0x0	rtc RW 0x0		osf RW 0x0	sr RW 0x0	reserved_0 RO 0x0
dmagrp_interrupt_enable 0x101C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_17 RO 0x0															nie RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	aie RW 0x0	ere RW 0x0	fbe RW 0x0	reserved_12_11 RO 0x0		ete RW 0x0	rwe RW 0x0	rse RW 0x0	rue RW 0x0	rie RW 0x0	une RW 0x0	ove RW 0x0	tje RW 0x0	tue RW 0x0	tse RW 0x0	tie RW 0x0
dmagrp_missed_frame_and_buffer_overflow_counter 0x1020	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_29 RO 0x0			ovfcntovf RO 0x0	ovffrmcnt RO 0x0											miscntovf RO 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	misfrmcnt RO 0x0															
dmagrp_receive_interrupt_watchdog_timer 0x1024	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_8 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved_31_8 RO 0x0								riwt RW 0x0							

Register Address Offset	Bit Fields															
dmagrp_axi_bus_mode 0x1028	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	en_lpi RW 0x0	lpi_xit_frm RW 0x0	Reserved						wr_osr_lmt RW 0x1				rd_osr_lmt RW 0x1			
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		onekbbe RW 0x0	axi_aal RO 0x0	Reserved								blen16 RW 0x0	blen8 RW 0x0	blen4 RW 0x0	undefined RO 0x1
dmagrp_ahb_or_axi_status 0x102C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_2 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved_31_2 RO 0x0														axirdsts RO 0x0	axwhsts RO 0x0
dmagrp_current_host_transmit_descriptor 0x1048	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	curtdesaptr RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	curtdesaptr RO 0x0															
dmagrp_current_host_receive_descriptor 0x104C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	currdesaptr RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	currdesaptr RO 0x0															
dmagrp_current_host_transmit_buffer_address 0x1050	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	curtbufaptr RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	curtbufaptr RO 0x0															

Register Address Offset	Bit Fields															
dmagrp_current_host_receive_buffer_address 0x1054	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	currbufaptr RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	currbufaptr RO 0x0															
dmagrp_hw_feature 0x1058	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31 RO 0x0	actphyif RO 0x0			savlanins RO 0x1	flexippsen RO 0x0	inttsen RO 0x0	enhdessel RO 0x1	txchcnt RO 0x1		rxchcnt RO 0x0		rxfifosize RO 0x0	rxtyp2coe RO 0x1	rxtyp1coe RO 0x0	txoesel RO 0x1
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	avsel RO 0x1	eeesel RO 0x1	tsver2sel RO 0x1	tsver1sel RO 0x0	mmcsel RO 0x1	mgksel RO 0x1	rwksel RO 0x0	smasel RO 0x1	l3l4fltren RO 0x1	pcssel RO 0x0	addmacadrsel RO 0x1	hashsel RO 0x0	exthashen RO 0x0	hdsel RO 0x1	gmiisel RO 0x1	miisel RO 0x1
i_emac_emac2																
gmacgrp_mac_configuration 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31 RO 0x0	sarc RW 0x0			twokpe RW 0x0	sfterr RO 0x0	cst RW 0x0	tc RO 0x0	wd RW 0x0	jd RW 0x0	be RW 0x0	je RW 0x0	ifg RW 0x0			dcrs RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ps RW 0x0	fes RW 0x0	do RW 0x0	lm RW 0x0	dm RW 0x0	ipc RW 0x0	dr RW 0x0	lud RO 0x0	acs RW 0x0	bl RW 0x0		dc RW 0x0	te RW 0x0	re RW 0x0	prelen RW 0x0	
gmacgrp_mac_frame_filter 0x4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ra RW 0x0	reserved_30_22 RO 0x0									dntu RW 0x0	ipfe RW 0x0	reserved_19_17 RO 0x0			vtfe RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved_15_11 RO 0x0					hpf RO 0x0	saf RW 0x0	saif RW 0x0	pcf RW 0x0		dbf RW 0x0	pm RW 0x0	daif RW 0x0	hmc RO 0x0	huc RO 0x0	pr RW 0x0

Register Address Offset	Bit Fields															
gmacgrp_gmii_address 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_16 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	pa RW 0x0					gr RW 0x0					cr RW 0x0				gw RW 0x0	gb RW 0x0
gmacgrp_gmii_data 0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_16 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	gd RW 0x0															
gmacgrp_flow_control 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	pt RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved_15_8 RO 0x0								dzpq RW 0x0	reserved_6 RO 0x0	plt RW 0x0		up RW 0x0	rfe RW 0x0	tfe RW 0x0	fca_bpa RW 0x0
gmacgrp_vlan_tag 0x1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_20 RO 0x0												vthm RO 0x0	esvl RW 0x0	vtim RW 0x0	etv RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	vl RW 0x0															
gmacgrp_version 0x20	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_16 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	userver RO 0x10								snpsver RO 0x37							

Register Address Offset	Bit Fields															
gmacgrp_debug 0x24	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_26 RO 0x0						txstsfsts RO 0x0	txfsts RO 0x0	reserved_23 RO 0x0	twcsts RO 0x0	trcsts RO 0x0		txpaused RO 0x0	tfcsts RO 0x0		tpests RO 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved_15_10 RO 0x0						rxfsts RO 0x0		reserved_7 RO 0x0	rrcsts RO 0x0		rwcsts RO 0x0	reserved_3 RO 0x0	rfcfcsts RO 0x0		rpests RO 0x0
gmacgrp_lpi_control_status 0x30	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_20 RO 0x0												lpitxa RW 0x0	plsen RO 0x0	pls RW 0x0	lpien RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved_15_10 RO 0x0						rlpist RO 0x0	tlpist RO 0x0	reserved_7_4 RO 0x0				rlpiex RO 0x0	rlpien RO 0x0	tlpiex RO 0x0	tlpien RO 0x0
gmacgrp_lpi_timers_control 0x34	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_26 RO 0x0						lst RW 0x3E8									
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	twt RW 0x0															
gmacgrp_interrupt_status 0x38	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_12 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved_31_12 RO 0x0				gpiis RO 0x0	lpiis RO 0x0	tsis RO 0x0	reserved_8 RO 0x0	mmcrxipis RO 0x0	mmctxis RO 0x0	mmcrxis RO 0x0	mmcis RO 0x0	pmtis RO 0x0	pcsancis RO 0x0	pcslchgis RO 0x0	rgsmiiis RO 0x0

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_11 RO 0x0															
gmacgrp_interrupt_mask 0x3C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved_31_11 RO 0x0					lpiim RW 0x0	tsim RO 0x0	reserved_8_4 RO 0x0					pmtim RW 0x0	pcsancim RO 0x0	pcslchgim RO 0x0	rgsmiiim RO 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gmacgrp_mac_address0_high 0x40	ae RO 0x1	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gmacgrp_mac_address0_low 0x44	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gmacgrp_mac_address1_high 0x48	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gmacgrp_mac_address1_low 0x4C	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address2_high 0x50	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address2_low 0x54	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address3_high 0x58	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address3_low 0x5C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address4_high 0x60	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address4_low 0x64	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address5_high 0x68	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address5_low 0x6C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address6_high 0x70	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address6_low 0x74	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gmacgrp_mac_address7_high 0x78	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gmacgrp_mac_address7_low 0x7C	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gmacgrp_mac_address8_high 0x80	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gmacgrp_mac_address8_low 0x84	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gmacgrp_mac_address9_high 0x88	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address9_low 0x8C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address10_high 0x90	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address10_low 0x94	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address11_high 0x98	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address11_low 0x9C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address12_high 0xA0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address12_low 0xA4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address13_high 0xA8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address13_low 0xAC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address14_high 0xB0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address14_low 0xB4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address15_high 0xB8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address15_low 0xBC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_sgmii_rgmii_smii_control_status 0xD8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved												lnksts RO 0x0	lnkspeed RO 0x0		lnkmod RO 0x0
gmacgrp_wdog_timeout 0xDC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_17 RO 0x0															pwe RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved_15_14 RO 0x0		wto RW 0x0													

Register Address Offset	Bit Fields															
gmacgrp_gen pio 0xE0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_x RO 0x0							gpit RW 0x0	reserved_23_x RO 0x0							gpie RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved_15_x RO 0x0							gpo RW 0x0	reserved_7_x RO 0x0							gpis RO 0x0
gmacgrp_mmc _control 0x100	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_9 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved_31_9 RO 0x0							ucdbc RW 0x0	reserved_7_6 RO 0x0		cntprstlvl RW 0x0	cntprst RW 0x0	cntfreez RW 0x0	rstonrd RW 0x0	cntstopro RW 0x0	cntrst RW 0x0
gmacgrp_mmc _receive_in terrupt 0x104	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_26 RO 0x0						rxctrlfis RO 0x0	rxrcverrfis RO 0x0	rxwdogfis RO 0x0	rxvlangbfis RO 0x0	rxfovfis RO 0x0	rxpausfis RO 0x0	rxorangefis RO 0x0	rxlenerfis RO 0x0	rxucgfis RO 0x0	rx1024tmaxoctgbfis RO 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rx512t1023octgbfis RO 0x0	rx256t511octgbfis RO 0x0	rx128t255octgbfis RO 0x0	rx65t127octgbfis RO 0x0	rx64octgbfis RO 0x0	rxosizegfis RO 0x0	rxusizegfis RO 0x0	rxjaberfis RO 0x0	rxruntfis RO 0x0	rxalignerfis RO 0x0	rxcrcerfis RO 0x0	rxmcgfis RO 0x0	rxbcgfis RO 0x0	rxgoctis RO 0x0	rxgboctis RO 0x0	rxgbfrmis RO 0x0
gmacgrp_mmc _transmit_i nterrupt 0x108	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_26 RO 0x0						txosizegfis RO 0x0	txvlangfis RO 0x0	txpausfis RO 0x0	txexdeffis RO 0x0	txgfrmis RO 0x0	txgoctis RO 0x0	txcarerfis RO 0x0	txexcolfis RO 0x0	txlatcolfis RO 0x0	txdeffis RO 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	txmcolgfis RO 0x0	txscolgfis RO 0x0	txuflowerfis RO 0x0	txbcgbfis RO 0x0	txmcgbfis RO 0x0	txucgbfis RO 0x0	tx1024tmaxoctgbfis RO 0x0	tx512t1023octgbfis RO 0x0	tx256t511octgbfis RO 0x0	tx128t255octgbfis RO 0x0	tx65t127octgbfis RO 0x0	tx64octgbfis RO 0x0	txmcgfis RO 0x0	txbcgfis RO 0x0	txgbfrmis RO 0x0	txgboctis RO 0x0

Register Address Offset	Bit Fields															
gmacgrp_mmc_receive_interrupt_mask 0x10C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_26 RO 0x0						rxctrlfim RW 0x0	rxrcverrfim RW 0x0	rxwdogfim RW 0x0	rxvlangbfim RW 0x0	rxfovfim RW 0x0	rxpausfim RW 0x0	rxorangefim RW 0x0	rxlenerfim RW 0x0	rxucgfim RW 0x0	rx1024tmaxoctgbfim RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rx512t1023octgbfim RW 0x0	rx256t511octgbfim RW 0x0	rx128t255octgbfim RW 0x0	rx65t127octgbfim RW 0x0	rx64octgbfim RW 0x0	rxosizegfim RW 0x0	rxusizegfim RW 0x0	rxjaberfim RW 0x0	rxruntfim RW 0x0	rxalignerfim RW 0x0	rxcrcerfim RW 0x0	rxmcgfim RW 0x0	rxbcgfim RW 0x0	rxgoctim RW 0x0	rxgboctim RW 0x0	rxgbfrmim RW 0x0
gmacgrp_mmc_transmit_interrupt_mask 0x110	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_26 RO 0x0						txosizegfim RW 0x0	txvlangfim RW 0x0	txpausfim RW 0x0	txexdeffim RW 0x0	txgfrmim RW 0x0	txgoctim RW 0x0	txcarerfim RW 0x0	txexcolfim RW 0x0	txlatcolfim RW 0x0	txdeffim RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	txmcolgfim RW 0x0	txscolgfim RW 0x0	txuflowerfim RW 0x0	txbcgbfim RW 0x0	txmcgbfim RW 0x0	txucgbfim RW 0x0	tx1024tmaxoctgbfim RW 0x0	tx512t1023octgbfim RW 0x0	tx256t511octgbfim RW 0x0	tx128t255octgbfim RW 0x0	tx65t127octgbfim RW 0x0	tx64octgbfim RW 0x0	txmcgfim RW 0x0	txbcgfim RW 0x0	txgbfrmim RW 0x0	txgboctim RW 0x0
gmacgrp_txoctetcount_gb 0x114	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_txframecount_gb 0x118	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															

Register Address Offset	Bit Fields															
gmacgrp_txbroadcast-frames_g 0x11C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_txmulticast-frames_g 0x120	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_tx64octets_gb 0x124	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_tx65to127octets_gb 0x128	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_tx128to255octets_gb 0x12C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_tx256to511octets_gb 0x130	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															

Register Address Offset	Bit Fields															
gmacgrp_tx512to1023octets_gb 0x134	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_tx1024tomaxoc-tets_gb 0x138	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_txunicastframes_gb 0x13C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_txmulticast-frames_gb 0x140	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_txbroadcast-frames_gb 0x144	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_txunderflo-werror 0x148	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															

Register Address Offset	Bit Fields															
gmacgrp_txsinglecol_g 0x14C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_txmulticol_g 0x150	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_txdeferred 0x154	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_txlatecol 0x158	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_txexesscol 0x15C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_txcarriererr 0x160	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															

Register Address Offset	Bit Fields															
gmacgrp_txoctetcnt 0x164	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	txoctetcount_g RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	txoctetcount_g RO 0x0															
gmacgrp_txframecount_g 0x168	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_txexcessdef 0x16C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_txpauseframes 0x170	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_txvlanframes_g 0x174	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_txoversize_g 0x178	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															

Register Address Offset	Bit Fields															
gmacgrp_rxframecount_gb 0x180	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxoctetcount_gb 0x184	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxoctetcount_g 0x188	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxbroadcast-frames_g 0x18C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxmulticast-frames_g 0x190	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxcrcerror 0x194	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															

Register Address Offset	Bit Fields															
gmacgrp_rxalignmenterror 0x198	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxrunterror 0x19C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxjabbererror 0x1A0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxundersize_g 0x1A4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxoversize_g 0x1A8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rx64octets_gb 0x1AC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															

Register Address Offset	Bit Fields															
gmacgrp_rx65to127octets_gb 0x1B0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rx128to255octets_gb 0x1B4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rx256to511octets_gb 0x1B8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rx512to1023octets_gb 0x1BC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rx1024tomaxoctets_gb 0x1C0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxunicastframes_g 0x1C4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															

Register Address Offset	Bit Fields															
gmacgrp_rxlengtherror 0x1C8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxoutofrange-type 0x1CC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxpauseframes 0x1D0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxfifooverflow 0x1D4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxvlanframes_gb 0x1D8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxwatchdogerror 0x1DC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															

Register Address Offset	Bit Fields															
gmacgrp_rxrcverror 0x1E0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxctrlframes_g 0x1E4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_mmc_ipc_receive_interrupt_mask 0x200	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved		rxicmperoim RW 0x0	rxicmpgoim RW 0x0	rxtcperoim RW 0x0	rxtcpgoim RW 0x0	rxudperoim RW 0x0	rxudpgoim RW 0x0	rxipv6nopayoim RW 0x0	rxipv6heroim RW 0x0	rxipv6goim RW 0x0	rxipv4udsbloim RW 0x0	rxipv4fragoim RW 0x0	rxipv4nopayoim RW 0x0	rxipv4heroim RW 0x0	rxipv4goim RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		rxicmperfim RW 0x0	rxicmpgfim RW 0x0	rxtcperfim RW 0x0	rxtcpgfim RW 0x0	rxudperfim RW 0x0	rxudpgfim RW 0x0	rxipv6nopayfim RW 0x0	rxipv6herfim RW 0x0	rxipv6gfim RW 0x0	rxipv4udsblfim RW 0x0	rxipv4fragfim RW 0x0	rxipv4nopayfim RW 0x0	rxipv4herfim RW 0x0	rxipv4gfim RW 0x0
gmacgrp_mmc_ipc_receive_interrupt 0x208	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved		rxicmperois RO 0x0	rxicmpgois RO 0x0	rxtcperois RO 0x0	rxtcpgois RO 0x0	rxudperois RO 0x0	rxudpgois RO 0x0	rxipv6nopayois RO 0x0	rxipv6herois RO 0x0	rxipv6gois RO 0x0	rxipv4udsblois RO 0x0	rxipv4fragois RO 0x0	rxipv4nopayois RO 0x0	rxipv4herois RO 0x0	rxipv4gois RO 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		rxicmperfis RO 0x0	rxicmpgfis RO 0x0	rxtcperfis RO 0x0	rxtcpgfis RO 0x0	rxudperfis RO 0x0	rxudpgfis RO 0x0	rxipv6nopayfis RO 0x0	rxipv6herfis RO 0x0	rxipv6gfis RO 0x0	rxipv4udsblfis RO 0x0	rxipv4fragfis RO 0x0	rxipv4nopayfis RO 0x0	rxipv4herfis RO 0x0	rxipv4gfis RO 0x0

Register Address Offset	Bit Fields															
gmacgrp_rxipv4_gd_frms 0x210	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxipv4_hdrerr_frms 0x214	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxipv4_nopay_frms 0x218	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxipv4_frag_frms 0x21C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxipv4_udsbl_frms 0x220	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxipv6_gd_frms 0x224	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															

Register Address Offset	Bit Fields															
gmacgrp_rxipv6_hdrerr_frms 0x228	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxipv6_nopay_frms 0x22C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxudp_gd_frms 0x230	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxudp_err_frms 0x234	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxtcp_gd_frms 0x238	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxtcp_err_frms 0x23C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															

Register Address Offset	Bit Fields															
gmacgrp_rxicmp_gd_frms 0x240	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxicmp_err_frms 0x244	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxipv4_gd_octets 0x250	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxipv4_hdrerr_octets 0x254	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxipv4_nopay_octets 0x258	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxipv4_frag_octets 0x25C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															

Register Address Offset	Bit Fields															
gmacgrp_rxipv4_udsbl_octets 0x260	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxipv6_gd_octets 0x264	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxipv6_hdrerr_octets 0x268	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxipv6_nopay_octets 0x26C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxudp_gd_octets 0x270	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxudp_err_octets 0x274	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															

Register Address Offset	Bit Fields															
gmacgrp_rxtcp_gd_octets 0x278	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxtcperroctets 0x27C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	rxtcp_err_octets RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rxtcp_err_octets RO 0x0															
gmacgrp_rxicmp_gd_octets 0x280	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_rxicmp_err_octets 0x284	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cnt RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cnt RO 0x0															
gmacgrp_l3_l4_control0 0x400	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_22 RO 0x0										l4dpim0 RW 0x0	l4dpm0 RW 0x0	l4spim0 RW 0x0	l4spm0 RW 0x0	reserved_17 RO 0x0	l4pen0 RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l3hdbm0 RW 0x0					l3hsbm0 RW 0x0					l3daim0 RW 0x0	l3dam0 RW 0x0	l3saim0 RW 0x0	l3sam0 RW 0x0	reserved_1 RO 0x0	l3pen0 RW 0x0

Register Address Offset	Bit Fields															
gmacgrp_layer4_address0 0x404	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	l4dp0 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l4sp0 RW 0x0															
gmacgrp_layer3_addr0_reg0 0x410	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	l3a00 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l3a00 RW 0x0															
gmacgrp_layer3_addr1_reg0 0x414	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	l3a10 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l3a10 RW 0x0															
gmacgrp_layer3_addr2_reg0 0x418	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	l3a20 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l3a20 RW 0x0															
gmacgrp_layer3_addr3_reg0 0x41C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	l3a30 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l3a30 RW 0x0															

Register Address Offset	Bit Fields															
gmacgrp_l3_l4_control1 0x430	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved										l4dpim1 RW 0x0	l4dpm1 RW 0x0	l4spim1 RW 0x0	l4spm1 RW 0x0	Reserved	l4pen1 RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l3hdbm1 RW 0x0					l3hsbm1 RW 0x0					l3daim1 RW 0x0	l3dam1 RW 0x0	l3saim1 RW 0x0	l3sam1 RW 0x0	Reserved	l3pen1 RW 0x0
gmacgrp_layer4_address1 0x434	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	l4dp1 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l4sp1 RW 0x0															
gmacgrp_layer3_addr0_reg1 0x440	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	l3a01 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l3a01 RW 0x0															
gmacgrp_layer3_addr1_reg1 0x444	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	l3a11 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l3a11 RW 0x0															
gmacgrp_layer3_addr2_reg1 0x448	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	l3a21 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l3a21 RW 0x0															

Register Address Offset	Bit Fields															
gmacgrp_layer3_addr3_reg1 0x44C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	l3a31 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l3a31 RW 0x0															
gmacgrp_l3_l4_control2 0x460	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved										l4dpim2 RW 0x0	l4dpm2 RW 0x0	l4spim2 RW 0x0	l4spm2 RW 0x0	Reserved	l4pen2 RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l3hdbm2 RW 0x0					l3hsbm2 RW 0x0					l3daim2 RW 0x0	l3dam2 RW 0x0	l3saim2 RW 0x0	l3sam2 RW 0x0	Reserved	l3pen2 RW 0x0
gmacgrp_layer4_address2 0x464	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	l4dp2 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l4sp2 RW 0x0															
gmacgrp_layer3_addr0_reg2 0x470	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	l3a02 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l3a02 RW 0x0															
gmacgrp_layer3_addr1_reg2 0x474	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	l3a12 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l3a12 RW 0x0															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gmacgrp_layer3_addr2_reg2 0x478	l3a22 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l3a22 RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gmacgrp_layer3_addr3_reg2 0x47C	l3a32 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l3a32 RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gmacgrp_l3_l4_control3 0x490	Reserved										l4dpim3 RW 0x0	l4dpm3 RW 0x0	l4spim3 RW 0x0	l4spm3 RW 0x0	Reserved	l4pen3 RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l3hdbm3 RW 0x0					l3hsbm3 RW 0x0					l3daim3 RW 0x0	l3dam3 RW 0x0	l3saim3 RW 0x0	l3sam3 RW 0x0	Reserved	l3pen3 RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gmacgrp_layer4_address3 0x494	l4dp3 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l4sp3 RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gmacgrp_layer3_addr0_reg3 0x4A0	l3a03 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l3a03 RW 0x0															

Register Address Offset	Bit Fields															
gmacgrp_layer3_addr1_reg3 0x4A4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	l3a13 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l3a13 RW 0x0															
gmacgrp_layer3_addr2_reg3 0x4A8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	l3a23 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l3a23 RW 0x0															
gmacgrp_layer3_addr3_reg3 0x4AC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	l3a33 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	l3a33 RW 0x0															
gmacgrp_hash_table_reg0 0x500	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ht31t0 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ht31t0 RW 0x0															
gmacgrp_hash_table_reg1 0x504	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ht63t32 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ht63t32 RW 0x0															
gmacgrp_hash_table_reg2 0x508	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ht95t64 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ht95t64 RW 0x0															

Register Address Offset	Bit Fields															
gmacgrp_hash_table_reg3 0x50C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ht127t96 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ht127t96 RW 0x0															
gmacgrp_hash_table_reg4 0x510	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ht159t128 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ht159t128 RW 0x0															
gmacgrp_hash_table_reg5 0x514	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ht191t160 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ht191t160 RW 0x0															
gmacgrp_hash_table_reg6 0x518	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ht223t196 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ht223t196 RW 0x0															
gmacgrp_hash_table_reg7 0x51C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ht255t224 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ht255t224 RW 0x0															
gmacgrp_vlan_incl_reg 0x584	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_20 RO 0x0												csvl RW 0x0	vlp RW 0x0	vlc RW 0x0	
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	vlt RW 0x0															

Register Address Offset	Bit Fields															
gmacgrp_vlan_hash_table_reg 0x588	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	vlht RW 0x0															
gmacgrp_timestamp_control 0x700	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_29 RO 0x0			atsen3 RO 0x0	atsen2 RO 0x0	atsen1 RO 0x0	atsen0 RO 0x0	atsfc RO 0x0	reserved_23_19 RO 0x0					tsenmacaddr RW 0x0	snaptypsel RW 0x0	
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	tsmstrena RW 0x0	tsevntena RW 0x0	tsipv4ena RW 0x1	tsipv6ena RW 0x0	tsipena RW 0x0	tsver2ena RW 0x0	tsctrlssr RW 0x0	tsenall RW 0x0	reserved_7_6 RO 0x0		tsaddreg RO 0x0	tstrig RO 0x0	tsupdt RO 0x0	tsinit RO 0x0	tscfupdt RO 0x0	tsena RW 0x0
gmacgrp_sub_second_increment 0x704	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								ssinc RW 0x0							
gmacgrp_system_time_seconds 0x708	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	tss RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	tss RO 0x0															
gmacgrp_system_time_nanoseconds 0x70C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	tsss RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	tsss RO 0x0															

Register Address Offset	Bit Fields															
gmacgrp_system_time_seconds_update 0x710	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	tss RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	tss RW 0x0															
gmacgrp_system_time_nanoseconds_update 0x714	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addsub RW 0x0	tsss RW 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	tsss RW 0x0															
gmacgrp_timestamp_addend 0x718	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	tsar RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	tsar RW 0x0															
gmacgrp_target_time_seconds 0x71C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	tstr RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	tstr RW 0x0															
gmacgrp_target_time_nanoseconds 0x720	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	trgtbusy RO 0x0	ttslo RW 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ttslo RW 0x0															
gmacgrp_system_time_higher_word_seconds 0x724	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	tshwr RW 0x0															

Register Address Offset	Bit Fields															
gmacgrp_timestamp_status 0x728	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved		atsns RO 0x0					atsstm RO 0x0	Reserved				atsstn RO 0x0			
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved												tstrgterr RO 0x0	auxtstrig RO 0x0	tstargt RO 0x0	tssovf RO 0x0
gmacgrp_pps_control 0x72C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved									trgtmodsel0 RW 0x0		ppsen0 RW 0x0	ppsctrl_ppscmd RW 0x0			
gmacgrp_auxiliary_timestamp_nanoseconds 0x730	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	auxtslo RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	auxtslo RO 0x0															
gmacgrp_auxiliary_timestamp_seconds 0x734	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	auxtshi RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	auxtshi RO 0x0															
gmacgrp_pps0_interval 0x760	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ppsint RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ppsint RW 0x0															

Register Address Offset	Bit Fields															
gmacgrp_pps0_width 0x764	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ppswidth RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ppswidth RW 0x0															
gmacgrp_mac_address16_high 0x800	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address16_low 0x804	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address17_high 0x808	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address17_low 0x80C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address18_high 0x810	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address18_low 0x814	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address19_high 0x818	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address19_low 0x81C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address20_high 0x820	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address20_low 0x824	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address21_high 0x828	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address21_low 0x82C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address22_high 0x830	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address22_low 0x834	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address23_high 0x838	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address23_low 0x83C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address24_high 0x840	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address24_low 0x844	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address25_high 0x848	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address25_low 0x84C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address26_high 0x850	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address26_low 0x854	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address27_high 0x858	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address27_low 0x85C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gmacgrp_mac_address28_high 0x860	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gmacgrp_mac_address28_low 0x864	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gmacgrp_mac_address29_high 0x868	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gmacgrp_mac_address29_low 0x86C	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gmacgrp_mac_address30_high 0x870	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address30_low 0x874	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address31_high 0x878	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address31_low 0x87C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address32_high 0x880	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address32_low 0x884	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address33_high 0x888	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address33_low 0x88C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address34_high 0x890	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address34_low 0x894	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address35_high 0x898	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gmacgrp_mac_address35_low 0x89C	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address36_high 0x8A0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address36_low 0x8A4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address37_high 0x8A8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address37_low 0x8AC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gmacgrp_mac_address38_high 0x8B0	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address38_low 0x8B4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address39_high 0x8B8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address39_low 0x8BC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address40_high 0x8C0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address40_low 0x8C4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address41_high 0x8C8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address41_low 0x8CC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address42_high 0x8D0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address42_low 0x8D4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gmacgrp_mac_address43_high 0x8D8	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address43_low 0x8DC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address44_high 0x8E0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address44_low 0x8E4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address45_high 0x8E8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address45_low 0x8EC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address46_high 0x8F0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address46_low 0x8F4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address47_high 0x8F8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address47_low 0x8FC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address48_high 0x900	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address48_low 0x904	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address49_high 0x908	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address49_low 0x90C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address50_high 0x910	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gmacgrp_mac_address50_low 0x914	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address51_high 0x918	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address51_low 0x91C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address52_high 0x920	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address52_low 0x924	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address53_high 0x928	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address53_low 0x92C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address54_high 0x930	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address54_low 0x934	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address55_high 0x938	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address55_low 0x93C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address56_high 0x940	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address56_low 0x944	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address57_high 0x948	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address57_low 0x94C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address58_high 0x950	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address58_low 0x954	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address59_high 0x958	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address59_low 0x95C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address60_high 0x960	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gmacgrp_mac_address60_low 0x964	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address61_high 0x968	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address61_low 0x96C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address62_high 0x970	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address62_low 0x974	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address63_high 0x978	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address63_low 0x97C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address64_high 0x980	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address64_low 0x984	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address65_high 0x988	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address65_low 0x98C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address66_high 0x990	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address66_low 0x994	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address67_high 0x998	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address67_low 0x99C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address68_high 0x9A0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address68_low 0x9A4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address69_high 0x9A8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address69_low 0x9AC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address70_high 0x9B0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address70_low 0x9B4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address71_high 0x9B8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address71_low 0x9BC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address72_high 0x9C0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address72_low 0x9C4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address73_high 0x9C8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address73_low 0x9CC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address74_high 0x9D0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address74_low 0x9D4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address75_high 0x9D8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address75_low 0x9DC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address76_high 0x9E0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address76_low 0x9E4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address77_high 0x9E8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address77_low 0x9EC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address78_high 0x9F0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address78_low 0x9F4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address79_high 0x9F8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address79_low 0x9FC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address80_high 0xA00	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address80_low 0xA04	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address81_high 0xA08	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address81_low 0xA0C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address82_high 0xA10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address82_low 0xA14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address83_high 0xA18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address83_low 0xA1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address84_high 0xA20	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address84_low 0xA24	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address85_high 0xA28	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address85_low 0xA2C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address86_high 0xA30	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address86_low 0xA34	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address87_high 0xA38	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address87_low 0xA3C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gmacgrp_mac_address88_high 0xA40	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address88_low 0xA44	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address89_high 0xA48	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address89_low 0xA4C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address90_high 0xA50	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address90_low 0xA54	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address91_high 0xA58	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address91_low 0xA5C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address92_high 0xA60	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address92_low 0xA64	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address93_high 0xA68	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address93_low 0xA6C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address94_high 0xA70	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address94_low 0xA74	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address95_high 0xA78	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address95_low 0xA7C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address96_high 0xA80	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address96_low 0xA84	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address97_high 0xA88	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address97_low 0xA8C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address98_high 0xA90	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address98_low 0xA94	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address99_high 0xA98	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address99_low 0xA9C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address100_high 0xAA0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address100_low 0xAA4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address101_high 0xAA8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address101_low 0xAAC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address102_high 0xAB0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address102_low 0xAB4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address103_high 0xAB8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address103_low 0xABC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address104_high 0xAC0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address104_low 0xAC4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address105_high 0xAC8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address105_low 0xACC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address106_high 0xAD0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address106_low 0xAD4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address107_high 0xAD8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address107_low 0xADC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address108_high 0xAE0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address108_low 0xAE4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address109_high 0xAE8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address109_low 0xAEC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address110_high 0xAF0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address110_low 0xAF4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address111_high 0xAF8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address111_low 0xAFC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address112_high 0xB00	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address112_low 0xB04	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address113_high 0xB08	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address113_low 0xB0C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address114_high 0xB10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address114_low 0xB14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address115_high 0xB18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address115_low 0xB1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address116_high 0xB20	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address116_low 0xB24	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address117_high 0xB28	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address117_low 0xB2C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address118_high 0xB30	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address118_low 0xB34	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address119_high 0xB38	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address119_low 0xB3C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address120_high 0xB40	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address120_low 0xB44	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address121_high 0xB48	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address121_low 0xB4C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address122_high 0xB50	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address122_low 0xB54	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address123_high 0xB58	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address123_low 0xB5C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address124_high 0xB60	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address124_low 0xB64	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address125_high 0xB68	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															

Register Address Offset	Bit Fields															
gmacgrp_mac_address125_low 0xB6C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address126_high 0xB70	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address126_low 0xB74	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															
gmacgrp_mac_address127_high 0xB78	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ae RW 0x0	reserved_30_16 RO 0x0														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrhi RW 0xFFFF															
gmacgrp_mac_address127_low 0xB7C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	addrlo RW 0xFFFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	addrlo RW 0xFFFFFFFF															

Register Address Offset	Bit Fields															
dmagrp_bus_mode 0x1000	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_26 RO 0x0						aal RW 0x0	eighxpbl RW 0x0	usp RW 0x0	rpbl RW 0x1						fb RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved_15_14 RO 0x0		pbl RW 0x1						atds RW 0x0	dsl RW 0x0					reserved_1 RO 0x0	swr RW 0x1
dmagrp_transmit_poll_demand 0x1004	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	tpd RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	tpd RW 0x0															
dmagrp_receive_poll_demand 0x1008	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	rpd RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rpd RW 0x0															
dmagrp_receive_descriptor_list_address 0x100C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	rdesla_32bit RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rdesla_32bit RW 0x0														Reserved	
dmagrp_transmit_descriptor_list_address 0x1010	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	tdesla_32bit RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	tdesla_32bit RW 0x0														Reserved	

Register Address Offset	Bit Fields															
dmagrp_status 0x1014	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31 RO 0x0	glpii RO 0x0	tti RO 0x0	gpi RO 0x0	gmi RO 0x0	gli RO 0x0	eb RO 0x0			ts RO 0x0			rs RO 0x0			nis RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ais RW 0x0	eri RW 0x0	fbi RW 0x0	reserved_12_11 RO 0x0		eti RW 0x0	rwt RW 0x0	rps RW 0x0	ru RW 0x0	ri RW 0x0	unf RW 0x0	ovf RW 0x0	tjt RW 0x0	tu RW 0x0	tps RW 0x0	ti RW 0x0
dmagrp_operation_mode 0x1018	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_27 RO 0x0					dt RW 0x0	rsf RW 0x0	dff RW 0x0	rfa_2 RO 0x0	rfd_2 RO 0x0	tsf RW 0x0	ftf RW 0x0	reserved_19_17 RO 0x0			ttc RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ttc RW 0x0		st RW 0x0	rfd RO 0x0		rfa RO 0x0		efc RO 0x0	fef RW 0x0	fuf RW 0x0	dgf RW 0x0	rtc RW 0x0		osf RW 0x0	sr RW 0x0	reserved_0 RO 0x0
dmagrp_interrupt_enable 0x101C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_17 RO 0x0															nie RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	aie RW 0x0	ere RW 0x0	fbe RW 0x0	reserved_12_11 RO 0x0		ete RW 0x0	rwe RW 0x0	rse RW 0x0	rue RW 0x0	rie RW 0x0	une RW 0x0	ove RW 0x0	tje RW 0x0	tue RW 0x0	tse RW 0x0	tie RW 0x0
dmagrp_missed_frame_and_buffer_overflow_counter 0x1020	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_29 RO 0x0			ovfcntovf RO 0x0	ovffrmcnt RO 0x0											miscntovf RO 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	misfrmcnt RO 0x0															
dmagrp_receive_interrupt_watchdog_timer 0x1024	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_8 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved_31_8 RO 0x0								riwt RW 0x0							

Register Address Offset	Bit Fields															
dmagrp_axi_bus_mode 0x1028	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	en_lpi RW 0x0	lpi_xit_frm RW 0x0	Reserved						wr_osr_lmt RW 0x1				rd_osr_lmt RW 0x1			
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		onekbbe RW 0x0	axi_aal RO 0x0	Reserved								blen16 RW 0x0	blen8 RW 0x0	blen4 RW 0x0	undefined RO 0x1
dmagrp_ahb_or_axi_status 0x102C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31_2 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved_31_2 RO 0x0														axirdsts RO 0x0	axwhsts RO 0x0
dmagrp_current_host_transmit_descriptor 0x1048	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	curtdesaptr RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	curtdesaptr RO 0x0															
dmagrp_current_host_receive_descriptor 0x104C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	currdesaptr RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	currdesaptr RO 0x0															
dmagrp_current_host_transmit_buffer_address 0x1050	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	curtbufaptr RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	curtbufaptr RO 0x0															

Register Address Offset	Bit Fields															
dmagrp_current_host_receive_buffer_address 0x1054	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	currbufaptr RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	currbufaptr RO 0x0															
dmagrp_hw_feature 0x1058	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved_31 RO 0x0	actphyif RO 0x0			savlanins RO 0x0	flexippsen RO 0x1	inttsen RO 0x1	enhdessel RO 0x1	txchcnt RO 0x0		rxchcnt RO 0x0		rxfifosize RO 0x0	rxtyp2coe RO 0x1	rxtyp1coe RO 0x0	txoesel RO 0x1
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	avsel RO 0x1	eeesel RO 0x1	tsver2sel RO 0x1	tsver1sel RO 0x0	mmcsel RO 0x1	mgksel RO 0x0	rwksel RO 0x0	smasel RO 0x1	l3l4fltren RO 0x1	pcssel RO 0x0	addmacadrsel RO 0x1	hashsel RO 0x0	exthashen RO 0x0	hdsel RO 0x1	gmiisel RO 0x1	miisel RO 0x1

gmacgrp_mac_configuration

Register 0 (MAC Configuration Register)

The MAC Configuration register establishes receive and transmit operating modes.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800000
i_emac_emac1	0xFF802000	0xFF802000
i_emac_emac2	0xFF804000	0xFF804000

Offset: 0x0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
reserved_31_28 RO 0x0				twokpe RW 0x0	sfterr RO 0x0	cst RW 0x0	tc RO 0x0	wd RW 0x0	jd RW 0x0	be RW 0x0	je RW 0x0	ifg RW 0x0			dcrs RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ps RW 0x0	fes RW 0x0	do RW 0x0	lm RW 0x0	dm RW 0x0	ipc RW 0x0	dr RW 0x0	lud RO 0x0	acs RW 0x0	bl RW 0x0		dc RW 0x0	te RW 0x0	re RW 0x0	prelen RW 0x0	

gmacgrp_mac_configuration Fields

Bit	Name	Description	Access	Reset
31	reserved_31_28	Reserved	RO	0x0
27	twokpe	IEEE 802.3as Support for 2K Packets When set, the MAC considers all frames, with up to 2,000 bytes length, as normal packets. When Bit 20 (JE) is not set, the MAC considers all received frames of size more than 2K bytes as Giant frames. When this bit is reset and Bit 20 (JE) is not set, the MAC considers all received frames of size more than 1,518 bytes (1,522 bytes for tagged) as Giant frames. When Bit 20 is set, setting this bit has no effect on Giant Frame status.	RW	0x0
26	sfterr	SMII Force Transmit Error When set, this bit indicates to the PHY to force a transmit error in the SMII frame being transmitted.	RO	0x0

Bit	Name	Description	Access	Reset
25	cst	CRC Stripping for Type Frames When this bit is set, the last 4 bytes (FCS) of all frames of Ether type (Length/Type field greater than or equal to 1,536) are stripped and dropped before forwarding the frame to the application. This function is not valid when the IP Checksum Engine (Type 1) is enabled in the MAC receiver. This function is valid when Type 2 Checksum Offload Engine is enabled. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
24	tc	Transmit Configuration in RGMII, SGMII, or SMII When set, this bit enables the transmission of duplex mode, link speed, and link up or down information to the PHY in the RGMII, SMII, or SGMII port. When this bit is reset, no such information is driven to the PHY. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RO	0x0

Bit	Name	Description	Access	Reset
23	wd	Watchdog Disable When this bit is set, the MAC disables the watchdog timer on the receiver. The MAC can receive frames of up to 16,384 bytes. When this bit is reset, the MAC does not allow a receive frame which more than 2,048 bytes (10,240 if JE is set high) or the value programmed in Register 55 (Watchdog Timeout Register). The MAC cuts off any bytes received after the watchdog limit number of bytes. **Value** **Description** 0x0 ENABLED 0x1 DISABLED	RW	0x0
22	jd	Jabber Disable When this bit is set, the MAC disables the jabber timer on the transmitter. The MAC can transfer frames of up to 16,384 bytes. When this bit is reset, the MAC cuts off the transmitter if the application sends out more than 2,048 bytes of data (10,240 if JE is set high) during transmission. **Value** **Description** 0x0 ENABLED 0x1 DISABLED	RW	0x0

Bit	Name	Description	Access	Reset
21	be	Frame Burst Enable When this bit is set, the MAC allows frame bursting during transmission in the GMII half-duplex mode. This bit is reserved (and RO) in the 10/100 Mbps only or full-duplex-only configurations. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
20	je	Jumbo Frame Enable When this bit is set, the MAC allows Jumbo frames of 9,018 bytes (9,022 bytes for VLAN tagged frames) without reporting a giant frame error in the receive frame status. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
19:17	ifg	Inter-Frame Gap These bits control the minimum IFG between frames during transmission. * 000: 96 bit times * 001: 88 bit times * 010: 80 bit times * ... * 111: 40 bit times In the half-duplex mode, the minimum IFG can be configured only for 64 bit times (IFG = 100). Lower values are not considered. In the 1000-Mbps mode, the minimum IFG supported is 64 bit times (and above). **Value** — **Description** 0x0 — IFG96BITTIMES 0x1 — IFG88BITTIMES 0x2 — IFG80BITTIMES 0x3 — IFG72BITTIMES 0x4 — IFG64BITTIMES 0x5 — IFG56BITTIMES 0x6 — IFG48BITTIMES 0x7 — IFG40BITTIMES	RW	0x0

Bit	Name	Description	Access	Reset
16	dcrs	Disable Carrier Sense During Transmission When set high, this bit makes the MAC transmitter ignore the (G)MII CRS signal during frame transmission in the half-duplex mode. This request results in no errors generated because of Loss of Carrier or No Carrier during such transmission. When this bit is low, the MAC transmitter generates such errors because of Carrier Sense and can even abort the transmissions. This bit is reserved (and RO) in the full-duplex-only configurations. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
15	ps	Port Select This bit selects the Ethernet line speed: * 0: For 1000 Mbps operations * 1: For 10 or 100 Mbps operations In 10 or 100 Mbps operations, this bit, along with FES bit, selects the exact line speed. In the 10 or 100 Mbps-only (always 1) or 1000 Mbps-only (always 0) configurations, this bit is read-only with the appropriate value. In default 10, 100, or 1000 Mbps configuration, this bit is R_W. The mac_portselect_o signal reflects the value of this bit. **Value** **Description** 0x0 GMII1000SEL 0x1 MII10100SEL	RW	0x0

Bit	Name	Description	Access	Reset
14	fes	Speed This bit selects the speed in the MII, RMII, SMII, RGMII, SGMII, or RevMII interface * 0: 10 Mbps * 1: 100 Mbps This bit is reserved (RO) by default and is enabled only when the parameter SPEED_SELECT = Enabled. This bit generates link speed encoding when Bit 24 (TC) is set in the RGMII, SMII, or SGMII mode. This bit is always enabled for RGMII, SGMII, SMII, or RevMII interface. In configurations with RGMII, SGMII, SMII, or RevMII interface, this bit is driven as an output signal (mac_speed_o[0]) to reflect the value of this bit in the mac_speed_o signal. In configurations with RMII, MII, or GMII interface, you can optionally drive this bit as an output signal (mac_speed_o[0]) to reflect its value in the mac_speed_o signal. **Value** — **Description** 0x0 — SPEED10 0x1 — SPEED100	RW	0x0

Bit	Name	Description	Access	Reset
13	do	Disable Receive Own When this bit is set, the MAC disables the reception of frames when the phy_txen_o is asserted in the half-duplex mode. When this bit is reset, the MAC receives all packets that are given by the PHY while transmitting. This bit is not applicable if the MAC is operating in the full-duplex mode. This bit is reserved (RO with default value) if the MAC is configured for the full-duplex-only operation. **Value** **Description** 0x0 ENABLED 0x1 DISABLED	RW	0x0
12	lm	Loopback Mode When this bit is set, the MAC operates in the loopback mode at GMII or MII. The (G)MII Receive clock input (clk_rx_i) is required for the loopback to work properly, because the Transmit clock is not looped-back internally. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
11	dm	Duplex Mode When this bit is set, the MAC operates in the full-duplex mode where it can transmit and receive simultaneously. This bit is RO with default value of 1'b1 in the full-duplex-only configuration. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
10	ipc	Checksum Offload When this bit is set, the MAC calculates the 16-bit one's complement of the one's complement sum of all received Ethernet frame payloads. It also checks whether the IPv4 Header checksum (assumed to be bytes 2526 or 2930 (VLAN-tagged) of the received Ethernet frame) is correct for the received frame and gives the status in the receive status word. The MAC also appends the 16-bit checksum calculated for the IP header datagram payload (bytes after the IPv4 header) and appends it to the Ethernet frame transferred to the application (when Type 2 COE is deselected). When this bit is reset, this function is disabled. When Type 2 COE is selected, this bit, when set, enables the IPv4 header checksum checking and IPv4 or IPv6 TCP, UDP, or ICMP payload checksum checking. When this bit is reset, the COE function in the receiver is disabled and the corresponding PCE and IP HCE status bits are always cleared. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
9	dr	Disable Retry When this bit is set, the MAC attempts only one transmission. When a collision occurs on the GMII or MII interface, the MAC ignores the current frame transmission and reports a Frame Abort with excessive collision error in the transmit frame status. When this bit is reset, the MAC attempts retries based on the settings of the BL field (Bits [6:5]). This bit is applicable only in the half-duplex mode and is reserved (RO with default value) in the full-duplex-only configuration. **Value** **Description** 0x0 ENABLED 0x1 DISABLED	RW	0x0
8	lud	Link Up or Down This bit indicates whether the link is up or down during the transmission of configuration in the RGMII, SGMII, or SMII interface: * 0: Link Down * 1: Link Up **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RO	0x0

Bit	Name	Description	Access	Reset
7	acs	Automatic Pad or CRC Stripping When this bit is set, the MAC strips the Pad or FCS field on the incoming frames only if the value of the length field is less than 1,536 bytes. All received frames with length field greater than or equal to 1,536 bytes are passed to the application without stripping the Pad or FCS field. When this bit is reset, the MAC passes all incoming frames, without modifying them, to the Host. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
6:5	bl	Back-Off Limit The Back-Off limit determines the random integer number (r) of slot time delays (4,096 bit times for 1000 Mbps and 512 bit times for 10/100 Mbps) for which the MAC waits before rescheduling a transmission attempt during retries after a collision. This bit is applicable only in the half-duplex mode and is reserved (RO) in the full-duplex-only configuration. * 00: k = min (n, 10) * 01: k = min (n, 8) * 10: k = min (n, 4) * 11: k = min (n, 1) where n = retransmission attempt. The random integer r takes the value in the range 0 <= r < kth power of 2 **Value** — **Description** 0x0 — BACKLIMTR10 0x1 — BACKLIMIRT8 0x2 — BACKLIMITR4 0x3 — BACKLIMITR1	RW	0x0

Bit	Name	Description	Access	Reset
4	dc	Deferral Check When this bit is set, the deferral check function is enabled in the MAC. The MAC issues a Frame Abort status, along with the excessive deferral error bit set in the transmit frame status, when the transmit state machine is deferred for more than 24,288 bit times in the 10 or 100 Mbps mode. If the MAC is configured for 1000 Mbps operation or if the Jumbo frame mode is enabled in the 10 or 100 Mbps mode, the threshold for deferral is 155,680 bits times. Deferral begins when the transmitter is ready to transmit, but it is prevented because of an active carrier sense signal (CRS) on GMII or MII. The defer time is not cumulative. For example, if the transmitter defers for 10,000 bit times because the CRS signal is active and then the CRS signal becomes inactive, the transmitter transmits and collision happens. Because of collision, the transmitter needs to back off and then defer again after back off completion. In such a scenario, the deferral timer is reset to 0 and it is restarted. When this bit is reset, the deferral check function is disabled and the MAC defers until the CRS signal goes inactive. This bit is applicable only in the half-duplex mode and is reserved (RO) in the full-duplex-only configuration. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
3	te	Transmitter Enable When this bit is set, the transmit state machine of the MAC is enabled for transmission on the GMII or MII. When this bit is reset, the MAC transmit state machine is disabled after the completion of the transmission of the current frame, and does not transmit any further frames. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
2	re	Receiver Enable When this bit is set, the receiver state machine of the MAC is enabled for receiving frames from the GMII or MII. When this bit is reset, the MAC receive state machine is disabled after the completion of the reception of the current frame, and does not receive any further frames from the GMII or MII. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
1:0	prelen	Preamble Length for Transmit Frames These bits control the number of preamble bytes that are added to the beginning of every Transmit frame. The preamble reduction occurs only when the MAC is operating in the full-duplex mode. * 2'b00: 7 bytes of preamble * 2'b01: 5 byte of preamble * 2'b10: 3 bytes of preamble * 2'b11: 1 byte of preamble **Value** **Description** 0x0 PREAM7BYTES 0x1 PREAM5BYTES 0x2 PREAM3BYTES 0x3 PREAM1BYTE	RW	0x0

gmacgrp_mac_frame_filter

Register 1 (MAC Frame Filter)

The MAC Frame Filter register contains the filter controls for receiving frames. Some of the controls from this register go to the address check block of the MAC, which performs the first level of address filtering. The second level of filtering is performed on the incoming frame, based on other controls such as Pass Bad Frames and Pass Control Frames.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800004
i_emac_emac1	0xFF802000	0xFF802004
i_emac_emac2	0xFF804000	0xFF804004

Offset: 0x4

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ra RW 0x0	reserved_30_22 RO 0x0									dntu RW 0x0	ipfe RW 0x0	reserved_19_17 RO 0x0			vtfe RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved_15_11 RO 0x0					hpf RO 0x0	saf RW 0x0	saif RW 0x0	pcf RW 0x0		dbf RW 0x0	pm RW 0x0	daif RW 0x0	hmc RO 0x0	huc RO 0x0	pr RW 0x0

gmacgrp_mac_frame_filter Fields

Bit	Name	Description	Access	Reset
31	ra	Receive All When this bit is set, the MAC Receiver module passes all received frames, irrespective of whether they pass the address filter or not, to the Application. The result of the SA or DA filtering is updated (pass or fail) in the corresponding bits in the Receive Status Word. When this bit is reset, the Receiver module passes only those frames to the Application that pass the SA or DA address filter. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:22	reserved_30_22	Reserved	RO	0x0

Bit	Name	Description	Access	Reset
21	dntu	Drop non-TCP/UDP over IP Frames When set, this bit enables the MAC to drop the non-TCP or UDP over IP frames. The MAC forward only those frames that are processed by the Layer 4 filter. When reset, this bit enables the MAC to forward all non-TCP or UDP over IP frames. **Value** **Description** 0x0 NODROP 0x1 DROP	RW	0x0
20	ipfe	Layer 3 and Layer 4 Filter Enable When set, this bit enables the MAC to drop frames that do not match the enabled Layer 3 and Layer 4 filters. If Layer 3 or Layer 4 filters are not enabled for matching, this bit does not have any effect. When reset, the MAC forwards all frames irrespective of the match status of the Layer 3 and Layer 4 fields. **Value** **Description** 0x0 NODROP 0x1 DROP	RW	0x0
19:17	reserved_19_17	Reserved	RO	0x0

Bit	Name	Description	Access	Reset
16	vtfe	VLAN Tag Filter Enable When set, this bit enables the MAC to drop VLAN tagged frames that do not match the VLAN Tag comparison. When reset, the MAC forwards all frames irrespective of the match status of the VLAN Tag. **Value** **Description** 0x0 NODROP 0x1 DROP	RW	0x0
15:11	reserved_15_11	Reserved	RO	0x0
10	hpf	Hash or Perfect Filter When this bit is set, it configures the address filter to pass a frame if it matches either the perfect filtering or the hash filtering as set by the HMC or HUC bits. When this bit is low and the HUC or HMC bit is set, the frame is passed only if it matches the Hash filter. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RO	0x0

Bit	Name	Description	Access	Reset
9	saf	Source Address Filter Enable When this bit is set, the MAC compares the SA field of the received frames with the values programmed in the enabled SA registers. If the comparison fails, the MAC drops the frame. When this bit is reset, the MAC forwards the received frame to the application with updated SAF bit of the Rx Status depending on the SA address comparison. Note: According to the IEEE specification, Bit 47 of the SA is reserved and set to 0. However, in DWC_gmac, the MAC compares all 48 bits. The software driver should take this into consideration while programming the MAC address registers for SA. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
8	saif	SA Inverse Filtering When this bit is set, the Address Check block operates in inverse filtering mode for the SA address comparison. The frames whose SA matches the SA registers are marked as failing the SA Address filter. When this bit is reset, frames whose SA does not match the SA registers are marked as failing the SA Address filter. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
7:6	pcf	Pass Control Frames These bits control the forwarding of all control frames (including unicast and multicast PAUSE frames).	RW	0x0

Bit	Name	Description	Access	Reset
		• 00: MAC filters all control frames from reaching the application. • 01: MAC forwards all control frames except PAUSE control frames to application even if they fail the Address filter. • 10: MAC forwards all control frames to application even if they fail the Address Filter. • 11: MAC forwards control frames that pass the Address Filter. The following conditions should be true for the PAUSE control frames processing: • Condition 1: The MAC is in the full-duplex mode and flow control is enabled by setting Bit 2 (RFE) of Register 6 (Flow Control Register) to 1. • Condition 2: The destination address (DA) of the received frame matches the special multicast address or the MAC Address 0 when Bit 3 (UP) of the Register 6 (Flow Control Register) is set. • Condition 3: The Type field of the received frame is 0x8808 and the OPCODE field is 0x0001. **Note:** This field should be set to 01 only when the Condition 1 is true, that is, the MAC is programmed to operate in the full-duplex mode and the RFE bit is enabled. Otherwise, the PAUSE frame filtering may be inconsistent. When Condition 1 is false, the PAUSE frames are considered as generic control frames. Therefore, to pass all control frames (including PAUSE control frames) when the full-duplex mode and flow control is not enabled, you should set the PCF field to 10 or 11 (as required by the application). **Value** — **Description** 0x0 — MACFLTALLCFR 0x1 — MACFWDXPAUSE 0x2 — MACFWDFAIL 0x3 — MACFWDPASS		

Bit	Name	Description	Access	Reset
5	dbf	Disable Broadcast Frames When this bit is set, the AFM module filters all incoming broadcast frames. In addition, it overrides all other filter settings. When this bit is reset, the AFM module passes all received broadcast frames. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
4	pm	Pass All Multicast When set, this bit indicates that all received frames with a multicast destination address (first bit in the destination address field is '1') are passed. When reset, filtering of multicast frame depends on HMC bit. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
3	daif	DA Inverse Filtering When this bit is set, the Address Check block operates in inverse filtering mode for the DA address comparison for both unicast and multicast frames. When reset, normal filtering of frames is performed. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
2	hmc	Hash Multicast When set, MAC performs destination address filtering of received multicast frames according to the hash table. When reset, the MAC performs a perfect destination address filtering for multicast frames, that is, it compares the DA field with the values programmed in DA registers. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RO	0x0
1	huc	Hash Unicast When set, MAC performs destination address filtering of unicast frames according to the hash table. When reset, the MAC performs a perfect destination address filtering for unicast frames, that is, it compares the DA field with the values programmed in DA registers. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RO	0x0

Bit	Name	Description	Access	Reset
0	pr	Promiscuous Mode When this bit is set, the Address Filter module passes all incoming frames regardless of its destination or source address. The SA or DA Filter Fails status bits of the Receive Status Word are always cleared when PR is set. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0

gmacgrp_gmii_address

Register 4 (GMII Address Register)

The GMII Address register controls the management cycles to the external PHY through the management interface.

Note: This register is present for all PHY interface when you select the Station Management (MDIO) feature in coreConsultant.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800010
i_emac_emac1	0xFF802000	0xFF802010
i_emac_emac2	0xFF804000	0xFF804010

Offset: 0x10

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
reserved_31_16 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
pa RW 0x0					gr RW 0x0					cr RW 0x0				gw RW 0x0	gb RW 0x0

gmacgrp_gmii_address Fields

Bit	Name	Description	Access	Reset
31:16	reserved_31_16	Reserved	RO	0x0
15:11	pa	Physical Layer Address This field indicates which of the 32 possible PHY devices are being accessed. For RevMII, this field gives the PHY Address of the RevMII module.	RW	0x0
10:6	gr	GMII Register These bits select the desired GMII register in the selected PHY device. For RevMII, these bits select the desired CSR register in the RevMII Registers set.	RW	0x0
5:2	cr	CSR Clock Range The CSR Clock Range selection determines the frequency of the MDC clock according to the CSR clock frequency used in your design. The suggested range of CSR clock frequency applicable for each value (when Bit[5] = 0) ensures that the MDC clock is approximately between the frequency range 1.0 MHz - 2.5 MHz. • 0000: The frequency of the CSR clock is 60-100 MHz and the MDC clock is CSR clock/42. • 0001: The frequency of the CSR clock is 100-150 MHz and the MDC clock is CSR clock/62. • 0010: The frequency of the CSR clock is 20-35 MHz and the MDC clock is CSR clock/16. • 0011: The frequency of the CSR clock is 35-60 MHz and the MDC clock is CSR clock/26.	RW	0x0

Bit	Name	Description	Access	Reset
		• 0100: The frequency of the CSR clock is 150-250 MHz and the MDC clock is CSR clock/102. • 0100: The frequency of the CSR clock is 250-300 MHz and the MDC clock is CSR clock/124. • 0110 and 0111: Reserved When Bit 5 is set, you can achieve MDC clock of frequency higher than the IEEE 802.3 specified frequency limit of 2.5 MHz and program a clock divider of lower value. For example, when CSR clock is of 100 MHz frequency and you program these bits as 1010, then the resultant MDC clock is of 12.5 MHz which is outside the limit of IEEE 802.3 specified range. Program the following values only if the interfacing chips support faster MDC clocks: • 1000: CSR clock/4 • 1001: CSR clock/6 • 1010: CSR clock/8 • 1011: CSR clock/10 • 1100: CSR clock/12 • 1101: CSR clock/14 • 1110: CSR clock/16 • 1111: CSR clock/18 These bits are not used for accessing RevMII. These bits are read-only if the RevMII interface is selected as single PHY interface. **Value** — **Description** 0xa — DIV8 0xb — DIV10 0xc — DIV12 0xd — DIV14 0xe — DIV16AGAIN 0xf — DIV18 0x0 — DIV42 0x1 — DIV62 0x2 — DIV16 0x3 — DIV26 0x4 — DIV102 0x5 — DIV124 0x8 — DIV4		

Bit	Name	Description	Access	Reset
		Value **Description** 0x9 DIV6		
1	gw	GMII Write When set, this bit indicates to the PHY or RevMII that this is a Write operation using the GMII Data register. If this bit is not set, it indicates that this is a Read operation, that is, placing the data in the GMII Data register. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
0	gb	GMII Busy This bit should read logic 0 before writing to Register 4 and Register 5. During a PHY or RevMII register access, the software sets this bit to 1'b1 to indicate that a Read or Write access is in progress. Register 5 is invalid until this bit is cleared by the MAC. Therefore, Register 5 (GMII Data) should be kept valid until the MAC clears this bit during a PHY Write operation. Similarly for a read operation, the contents of Register 5 are not valid until this bit is cleared. The subsequent read or write operation should happen only after the previous operation is complete. Because there is no acknowledgment from the PHY to MAC after a read or write operation is completed, there is no change in the functionality of this bit even when the PHY is not present. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0

gmacgrp_gmii_data

Register 5 (GMII Data Register)

The GMII Data register stores Write data to be written to the PHY register located at the address specified in Register 4 (GMII Address Register). This register also stores the Read data from the PHY register located at the address specified by Register 4.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800014
i_emac_emac1	0xFF802000	0xFF802014

Module Instance	Base Address	Register Address
i_emac_emac2	0xFF804000	0xFF804014

Offset: 0x14

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
reserved_31_16 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
gd RW 0x0															

gmacgrp_gmii_data Fields

Bit	Name	Description	Access	Reset
31:16	reserved_31_16	Reserved	RO	0x0
15:0	gd	GMII Data This field contains the 16-bit data value read from the PHY or RevMII after a Management Read operation or the 16-bit data value to be written to the PHY or RevMII before a Management Write operation.	RW	0x0

gmacgrp_flow_control

```
Register 6 (Flow Control Register)

The Flow Control register controls the generation and reception of
the Control (Pause Command) frames by the MAC's Flow control module.
A Write to a register with the Busy bit set to '1' triggers the Flow
Control block to generate a Pause Control frame. The fields of the
control frame are selected as specified in the 802.3x specification,
and the Pause Time value from this register is used in the Pause Time
field of the control frame. The Busy bit remains set until the
control frame is transferred onto the cable. The Host must make sure
```

that the Busy bit is cleared before writing to the register.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800018
i_emac_emac1	0xFF802000	0xFF802018
i_emac_emac2	0xFF804000	0xFF804018

Offset: 0x18

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
pt RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved_15_8 RO 0x0								dzpq RW 0x0	reserved_6 RO 0x0	plt RW 0x0		up RW 0x0	rfe RW 0x0	tfe RW 0x0	fca_bpa RW 0x0

gmacgrp_flow_control Fields

Bit	Name	Description	Access	Reset
31:16	pt	Pause Time This field holds the value to be used in the Pause Time field in the transmit control frame. If the Pause Time bits is configured to be double-synchronized to the (G)MII clock domain, then consecutive writes to this register should be performed only after at least four clock cycles in the destination clock domain.	RW	0x0
15:8	reserved_15_8	Reserved	RO	0x0

Bit	Name	Description	Access	Reset
7	dzpq	Disable Zero-Quanta Pause When this bit is set, it disables the automatic generation of the Zero-Quanta Pause Control frames on the de-assertion of the flow-control signal from the FIFO layer (MTL or external sideband flow control signal sbd_flowctrl_i/mti_flowctrl_i). When this bit is reset, normal operation with automatic Zero-Quanta Pause Control frame generation is enabled. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
6	reserved_6	Reserved	RO	0x0

Bit	Name	Description	Access	Reset
5:4	plt	Pause Low Threshold This field configures the threshold of the PAUSE timer at which the input flow control signal mti_flowctrl_i (or sbd_flowctrl_i) is checked for automatic retransmission of PAUSE Frame. The threshold values should be always less than the Pause Time configured in Bits[31:16]. For example, if PT = 100H (256 slot-times), and PLT = 01, then a second PAUSE frame is automatically transmitted if the mti_flowctrl_i signal is asserted at 228 (256 - 28) slot times after the first PAUSE frame is transmitted. The following list provides the threshold values for different values: - 00: The threshold is Pause time minus 4 slot times (PT - 4 slot times). - 01: The threshold is Pause time minus 28 slot times (PT - 28 slot times). - 10: The threshold is Pause time minus 144 slot times (PT - 144 slot times). - 11: The threshold is Pause time minus 256 slot times (PT - 256 slot times). The slot time is defined as the time taken to transmit 512 bits (64 bytes) on the GMII or MII interface. **Value** **Description** 0x0 PAUSETIME_4 0x1 PAUSETIME_28 0x2 PAUSETIME_144 0x3 PAUSETIME_256	RW	0x0

Bit	Name	Description	Access	Reset
3	up	Unicast Pause Frame Detect A pause frame is processed when it has the unique multicast address specified in the IEEE Std 802.3. When this bit is set, the MAC can also detect Pause frames with unicast address of the station. This unicast address should be as specified in the MAC Address0 High Register and MAC Address0 Low Register. When this bit is reset, the MAC only detects Pause frames with unique multicast address. Note: The MAC does not process a Pause frame if the multicast address of received frame is different from the unique multicast address. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
2	rfe	Receive Flow Control Enable When this bit is set, the MAC decodes the received Pause frame and disables its transmitter for a specified (Pause) time. When this bit is reset, the decode function of the Pause frame is disabled. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
1	tfe	Transmit Flow Control Enable In the full-duplex mode, when this bit is set, the MAC enables the flow control operation to transmit Pause frames. When this bit is reset, the flow control operation in the MAC is disabled, and the MAC does not transmit any Pause frames. In half-duplex mode, when this bit is set, the MAC enables the back-pressure operation. When this bit is reset, the back-pressure feature is disabled. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
0	fca_bpa	Flow Control Busy or Backpressure Activate This bit initiates a Pause Control frame in the full-duplex mode and activates the backpressure function in the half-duplex mode if the TFE bit is set. In the full-duplex mode, this bit should be read as 1'b0 before writing to the Flow Control register. To initiate a Pause control frame, the Application must set this bit to 1'b1. During a transfer of the Control Frame, this bit continues to be set to signify that a frame transmission is in progress. After the completion of Pause control frame transmission, the MAC resets this bit to 1'b0. The Flow Control register should not be written to until this bit is cleared. In the half-duplex mode, when this bit is set (and TFE is set), then backpressure is asserted by the MAC. During backpressure, when the MAC receives a new frame, the transmitter starts sending a JAM pattern resulting in a collision. This control register bit is logically ORed with the mti_flowctrl_i input signal for the backpressure function. When the MAC is configured for the full-duplex mode, the BPA is automatically disabled. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0

gmacgrp_vlan_tag

Register 7 (VLAN Tag Register)

The VLAN Tag register contains the IEEE 802.1Q VLAN Tag to identify

the VLAN frames. The MAC compares the 13th and 14th bytes of the receiving frame (Length/Type) with 16'h8100, and the following two bytes are compared with the VLAN tag. If a match occurs, the MAC sets the received VLAN bit in the receive frame status. The legal length of the frame is increased from 1,518 bytes to 1,522 bytes.
If the VLAN Tag register is configured to be double-synchronized to the (G)MII clock domain, then consecutive writes to these register should be performed only after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF80001C
i_emac_emac1	0xFF802000	0xFF80201C
i_emac_emac2	0xFF804000	0xFF80401C

Offset: 0x1C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
reserved_31_20 RO 0x0												vthm RO 0x0	esvl RW 0x0	vtim RW 0x0	etv RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
vl RW 0x0															

gmacgrp_vlan_tag Fields

Bit	Name	Description	Access	Reset
31:20	reserved_31_20	Reserved	RO	0x0

Bit	Name	Description	Access	Reset
19	vthm	VLAN Tag Hash Table Match Enable When set, the most significant four bits of the VLAN tag's CRC are used to index the content of Register 354 (VLAN Hash Table Register). A value of 1 in the VLAN Hash Table register, corresponding to the index, indicates that the frame matched the VLAN hash table. When Bit 16 (ETV) is set, the CRC of the 12-bit VLAN Identifier (VID) is used for comparison whereas when ETV is reset, the CRC of the 16-bit VLAN tag is used for comparison. When reset, the VLAN Hash Match operation is not performed.	RO	0x0
18	esvl	Enable S-VLAN When this bit is set, the MAC transmitter and receiver also consider the S-VLAN (Type = 0x88A8) frames as valid VLAN tagged frames.	RW	0x0
17	vtim	VLAN Tag Inverse Match Enable When set, this bit enables the VLAN Tag inverse matching. The frames that do not have matching VLAN Tag are marked as matched. When reset, this bit enables the VLAN Tag perfect matching. The frames with matched VLAN Tag are marked as matched.	RW	0x0

Bit	Name	Description	Access	Reset
16	etv	Enable 12-Bit VLAN Tag Comparison When this bit is set, a 12-bit VLAN identifier is used for comparing and filtering instead of the complete 16-bit VLAN tag. Bits [11:0] of VLAN tag are compared with the corresponding field in the received VLAN-tagged frame. Similarly, when enabled, only 12 bits of the VLAN tag in the received frame are used for hash-based VLAN filtering. When this bit is reset, all 16 bits of the 15th and 16th bytes of the received VLAN frame are used for comparison and VLAN hash filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
15:0	vl	VLAN Tag Identifier for Receive Frames This field contains the 802.1Q VLAN tag to identify the VLAN frames and is compared to the 15th and 16th bytes of the frames being received for VLAN frames. The following list describes the bits of this field: * Bits [15:13]: User Priority * Bit 12: Canonical Format Indicator (CFI) or Drop Eligible Indicator (DEI) * Bits[11:0]: VLAN tag's VLAN Identifier (VID) field When the ETV bit is set, only the VID (Bits[11:0]) is used for comparison. If VL (VL[11:0] if ETV is set) is all zeros, the MAC does not check the fifteenth and 16th bytes for VLAN tag comparison, and declares all frames with a Type field value of 0x8100 or 0x88a8 as VLAN frames.	RW	0x0

gmacgrp_version

Register 8 (Version Register)

The Version registers identifies the version of the DWC_gmac.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800020
i_emac_emac1	0xFF802000	0xFF802020
i_emac_emac2	0xFF804000	0xFF804020

Offset: 0x20

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
reserved_31_16 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
userver RO 0x10								snpsver RO 0x37							

gmacgrp_version Fields

Bit	Name	Description	Access	Reset
31:16	reserved_31_16	Reserved	RO	0x0
15:8	userver	User-defined Version	RO	0x10
7:0	snpsver	Synopsys-defined Version (3.7)	RO	0x37

gmacgrp_debug

Register 9 (Debug Register)

The Debug register gives the status of all main modules of the transmit and receive data-paths and the FIFOs. An all-zero status

indicates that the MAC is in idle state (and FIFOs are empty) and no activity is going on in the data-paths.

Note:

The reset values, given for the Debug register, are valid only if the following clocks are present during the reset operation:
* clk_csr_i, clk_app_i, hclk_i, or aclk_i
* clk_tx_i
* clk_rx_i

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800024
i_emac_emac1	0xFF802000	0xFF802024
i_emac_emac2	0xFF804000	0xFF804024

Offset: 0x24

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
reserved_31_26 RO 0x0						txstsfsts RO 0x0	txfsts RO 0x0	reserved_23 RO 0x0	twcsts RO 0x0	trcsts RO 0x0		txpaused RO 0x0	tfcsts RO 0x0		tpests RO 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved_15_10 RO 0x0						rxfsts RO 0x0		reserved_7 RO 0x0	rrcsts RO 0x0		rwcsts RO 0x0	reserved_3 RO 0x0	rfcfcsts RO 0x0		rpests RO 0x0

gmacgrp_debug Fields

Bit	Name	Description	Access	Reset
31:26	reserved_31_26	Reserved	RO	0x0

Bit	Name	Description	Access	Reset
25	txstsfsts	MTL TxStatus FIFO Full Status When high, this bit indicates that the MTL TxStatus FIFO is full. Therefore, the MTL cannot accept any more frames for transmission. This bit is reserved in the GMAC-AHB and GMAC-DMA configurations. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0
24	txfsts	MTL Tx FIFO Not Empty Status When high, this bit indicates that the MTL Tx FIFO is not empty and some data is left for transmission. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0
23	reserved_23	Reserved	RO	0x0
22	twcsts	MTL Tx FIFO Write Controller Active Status When high, this bit indicates that the MTL Tx FIFO Write Controller is active and transferring data to the Tx FIFO. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0

Bit	Name	Description	Access	Reset
21:20	trcsts	MTL Tx FIFO Read Controller Status This field indicates the state of the Tx FIFO Read Controller: * 00: IDLE state * 01: READ state (transferring data to MAC transmitter) * 10: Waiting for TxStatus from MAC transmitter * 11: Writing the received TxStatus or flushing the Tx FIFO **Value** **Description** 0x0 IDLE 0x1 READSTATE 0x2 WAITTXSTAT 0x3 WRTXSTAT	RO	0x0
19	txpaused	MAC transmitter in PAUSE When high, this bit indicates that the MAC transmitter is in the PAUSE condition (in the full-duplex only mode) and hence does not schedule any frame for transmission. **Value** **Description** 0x0 DISABLE 0x1 ENABLED	RO	0x0

Bit	Name	Description	Access	Reset
18:17	tfcsts	MAC Transmit Frame Controller Status This field indicates the state of the MAC Transmit Frame Controller module: * 00: IDLE state * 01: Waiting for Status of previous frame or IFG or backoff period to be over * 10: Generating and transmitting a PAUSE control frame (in the full-duplex mode) * 11: Transferring input frame for transmission **Value** **Description** 0x0 IDLE 0x1 WAITIFG 0x2 XTPAUSE 0x3 XTINFRM	RO	0x0
16	tpests	MAC GMII or MII Transmit Protocol Engine Status When high, this bit indicates that the MAC GMII or MII transmit protocol engine is actively transmitting data and is not in the IDLE state. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RO	0x0
15:10	reserved_15_10	Reserved	RO	0x0

Bit	Name	Description	Access	Reset
9:8	rxfsts	MTL Rx FIFO Fill-level Status This field gives the status of the fill-level of the Rx FIFO: * 00: Rx FIFO Empty * 01: Rx FIFO fill level is below the flow-control deactivate threshold * 10: Rx FIFO fill level is above the flow-control activate threshold * 11: Rx FIFO Full **Value** **Description** 0x0 RXFIFOEMPTY 0x1 RXFIFOBELLVL 0x2 RXFIFOABLVL 0x3 RXFIFOFULL	RO	0x0
7	reserved_7	Reserved	RO	0x0
6:5	rrcsts	MTL Rx FIFO Read Controller State This field gives the state of the Rx FIFO read Controller: * 00: IDLE state * 01: Reading frame data * 10: Reading frame status (or timestamp) * 11: Flushing the frame data and status **Value** **Description** 0x0 IDLE 0x1 RDFRAMEDATA 0x2 RDFRAMESTAT 0x3 FLUSHFRDS	RO	0x0

Bit	Name	Description	Access	Reset
4	rwcsts	MTL Rx FIFO Write Controller Active Status When high, this bit indicates that the MTL Rx FIFO Write Controller is active and is transferring a received frame to the FIFO. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0
3	reserved_3	Reserved	RO	0x0
2:1	rfcfcsts	MAC Receive Frame Controller FIFO Status When high, this field indicates the active state of the small FIFO Read and Write controllers of the MAC Receive Frame Controller Module. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0
0	rpests	MAC GMII or MII Receive Protocol Engine Status When high, this bit indicates that the MAC GMII or MII receive protocol engine is actively receiving data and not in IDLE state. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0

gmacgrp_lpi_control_status

Register 12 (LPI Control and Status Register)

The LPI Control and Status Register controls the LPI functions and provides the LPI interrupt status. The status bits are cleared when

this register is read.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800030
i_emac_emac1	0xFF802000	0xFF802030
i_emac_emac2	0xFF804000	0xFF804030

Offset: 0x30

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
reserved_31_20 RO 0x0												lpitxa RW 0x0	plsen RO 0x0	pls RW 0x0	lpien RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved_15_10 RO 0x0						rlpist RO 0x0	tlpist RO 0x0	reserved_7_4 RO 0x0				rlpiex RO 0x0	rlpien RO 0x0	tlpiex RO 0x0	tlpien RO 0x0

gmacgrp_lpi_control_status Fields

Bit	Name	Description	Access	Reset
31:20	reserved_31_20	Reserved	RO	0x0

Bit	Name	Description	Access	Reset
19	lpitxa	LPI TX Automate This bit controls the behavior of the MAC when it is entering or coming out of the LPI mode on the transmit side. If the LPITXA and LPIEN bits are set to 1, the MAC enters the LPI mode only after all outstanding frames (in the core) and pending descriptors in system memory that the DMA must process have been transmitted. The MAC comes out of the LPI mode when the application sends any frame for transmission or the application issues a TX FIFO Flush command. In addition, the MAC automatically clears the LPIEN bit when it exits the LPI state. If TX FIFO Flush is set, in Bit 20 of Register 6 (Operation Mode Register), when the MAC is in the LPI mode, the MAC exits the LPI mode. When this bit is 0, the LPIEN bit directly controls behavior of the MAC when it is entering or coming out of the LPI mode. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
18	plsen	PHY Link Status Enable This bit enables the link status received on the RGMII, SGMII, or SMII receive paths to be used for activating the LPI LS TIMER. When set, the MAC uses the link-status bits of Register 54 (SGMII/RGMII/SMII Status Register) and Bit 17 (PLS) for the LPI LS Timer trigger. When cleared, the MAC ignores the link-status bits of Register 54 and takes only the PLS bit. This bit is RO and reserved if you have not selected the RGMII, SGMII, or SMII PHY interface. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RO	0x0
17	pls	PHY Link Status This bit indicates the link status of the PHY. The MAC Transmitter asserts the LPI pattern only when the link status is up (okay) at least for the time indicated by the LPI LS TIMER. When set, the link is considered to be okay (up) and when reset, the link is considered to be down. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
16	lpien	LPI Enable When set, this bit instructs the MAC Transmitter to enter the LPI state. When reset, this bit instructs the MAC to exit the LPI state and resume normal transmission. This bit is cleared when the LPITXA bit is set and the MAC exits the LPI state because of the arrival of a new packet for transmission. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
15:10	reserved_15_10	Reserved	RO	0x0
9	rlpist	Receive LPI State When set, this bit indicates that the MAC is receiving the LPI pattern on the GMII or MII interface. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0
8	tlpist	Transmit LPI State When set, this bit indicates that the MAC is transmitting the LPI pattern on the GMII or MII interface. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0
7:4	reserved_7_4	Reserved	RO	0x0

Bit	Name	Description	Access	Reset
3	rlpiex	Receive LPI Exit When set, this bit indicates that the MAC Receiver has stopped receiving the LPI pattern on the GMII or MII interface, exited the LPI state, and resumed the normal reception. This bit is cleared by a read into this register. Note: This bit may not get set if the MAC stops receiving the LPI pattern for a very short duration, such as, less than 3 clock cycles of CSR clock. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0
2	rlpien	Receive LPI Entry When set, this bit indicates that the MAC Receiver has received an LPI pattern and entered the LPI state. This bit is cleared by a read into this register. Note: This bit may not get set if the MAC stops receiving the LPI pattern for a very short duration, such as, less than 3 clock cycles of CSR clock. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0

Bit	Name	Description	Access	Reset
1	tlpiex	Transmit LPI Exit When set, this bit indicates that the MAC transmitter has exited the LPI state after the user has cleared the LPIEN bit and the LPI TW Timer has expired. This bit is cleared by a read into this register. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0
0	tlpien	Transmit LPI Entry When set, this bit indicates that the MAC Transmitter has entered the LPI state because of the setting of the LPIEN bit. This bit is cleared by a read into this register. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0

gmacgrp_lpi_timers_control

Register 13 (LPI Timers Control Register)

The LPI Timers Control register controls the timeout values in the LPI states. It specifies the time for which the MAC transmits the LPI pattern and also the time for which the MAC waits before resuming the normal transmission.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800034
i_emac_emac1	0xFF802000	0xFF802034
i_emac_emac2	0xFF804000	0xFF804034

Offset: 0x34

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
reserved_31_26 RO 0x0						lst RW 0x3E8									
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
twt RW 0x0															

gmacgrp_lpi_timers_control Fields

Bit	Name	Description	Access	Reset
31:26	reserved_31_26	Reserved	RO	0x0
25:16	lst	LPI LS Timer This field specifies the minimum time (in milliseconds) for which the link status from the PHY should be up (OKAY) before the LPI pattern can be transmitted to the PHY. The MAC does not transmit the LPI pattern even when the LPIEN bit is set unless the LPI LS Timer reaches the programmed terminal count. The default value of the LPI LS Timer is 1000 (1 sec) as defined in the IEEE standard.	RW	0x3E8
15:0	twt	LPI TW Timer This field specifies the minimum time (in microseconds) for which the MAC waits after it stops transmitting the LPI pattern to the PHY and before it resumes the normal transmission. The TLPIEX status bit is set after the expiry of this timer.	RW	0x0

gmacgrp_interrupt_status

Register 14 (Interrupt Register)

The Interrupt Status register identifies the events in the MAC that can generate interrupt. All interrupt events are generated only when the corresponding feature is enabled during operation. Therefore, these bits are reserved when the corresponding features are not present in the core.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800038
i_emac_emac1	0xFF802000	0xFF802038
i_emac_emac2	0xFF804000	0xFF804038

Offset: 0x38

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
reserved_31_12 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved_31_12 RO 0x0				gpiis RO 0x0	lpiis RO 0x0	tsis RO 0x0	reserved_8 RO 0x0	mmcrxipis RO 0x0	mmctxis RO 0x0	mmcrxis RO 0x0	mmcis RO 0x0	pmtis RO 0x0	pcsancis RO 0x0	pcslchgis RO 0x0	rgsmiiis RO 0x0

gmacgrp_interrupt_status Fields

Bit	Name	Description	Access	Reset
31:12	reserved_31_12	Reserved	RO	0x0

Bit	Name	Description	Access	Reset
11	gpiis	GPI Interrupt Status When the GPIO feature is enabled, this bit is set when any active event (LL or LH) occurs on GPIS field (Bits [3:0]) of Register 56 (General Purpose IO Register) and the corresponding GPIE bit is enabled. This bit is cleared on reading the lane 0 (GPIS) of Register 56 (Genaral Purpose IO Register). When the GPIO feature is not enabled, this bit is reserved.	RO	0x0
10	lpiis	LPI Interrupt Status When the Energy Efficient Ethernet feature is enabled, this bit is set for any LPI state entry or exit in the MAC Transmitter or Receiver. This bit is cleared on reading Bit 0 of Register 12 (LPI Control and Status Register). In all other modes, this bit is reserved. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0

Bit	Name	Description	Access	Reset
9	tsis	Timestamp Interrupt Status When the Advanced Timestamp feature is enabled, this bit is set when any of the following conditions is true: * The system time value equals or exceeds the value specified in the Target Time High and Low registers. * There is an overflow in the seconds register. * The Auxiliary snapshot trigger is asserted. This bit is cleared on reading Bit 0 of the Register 458 (Timestamp Status Register). If default Timestamping is enabled, when set, this bit indicates that the system time value is equal to or exceeds the value specified in the Target Time registers. In this mode, this bit is cleared after the completion of the read of this bit. In all other modes, this bit is reserved. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0
8	reserved_8	Reserved	RO	0x0
7	mmcrxipis	MMC Receive Checksum Offload Interrupt Status This bit is set high when an interrupt is generated in the MMC Receive Checksum Offload Interrupt Register. This bit is cleared when all the bits in this interrupt register are cleared. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0

Bit	Name	Description	Access	Reset
6	mmctxis	MMC Transmit Interrupt Status This bit is set high when an interrupt is generated in the MMC Transmit Interrupt Register. This bit is cleared when all the bits in this interrupt register are cleared. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0
5	mmcrxis	MMC Receive Interrupt Status This bit is set high when an interrupt is generated in the MMC Receive Interrupt Register. This bit is cleared when all the bits in this interrupt register are cleared. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RO	0x0
4	mmcis	MMC Interrupt Status This bit is set high when any of the Bits [7:5] is set high and cleared only when all of these bits are low. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0

Bit	Name	Description	Access	Reset
3	pmtis	PMT Interrupt Status This bit is set when a Magic packet or Wake-on-LAN frame is received in the power-down mode (see Bits 5 and 6 in the PMT Control and Status Register). This bit is cleared when both Bits[6:5] are cleared because of a read operation to the PMT Control and Status register.	RO	0x0
2	pcsancis	PCS Auto-Negotiation Complete This bit is set when the Auto-negotiation is completed in the TBI, RTBI, or SGMII PHY interface (Bit 5 in Register 49 (AN Status Register)). This bit is cleared when you perform a read operation to the AN Status register.	RO	0x0
1	pcslchgis	PCS Link Status Changed This bit is set because of any change in Link Status in the TBI, RTBI, or SGMII PHY interface (Bit 2 in Register 49 (AN Status Register)). This bit is cleared when you perform a read operation on the AN Status register.	RO	0x0
0	rgsmiiis	RGMII or SMII Interrupt Status This bit is set because of any change in value of the Link Status of RGMII or SMII interface (Bit 3 in Register 54 (SGMII/RGMII/SMII Status Register)). This bit is cleared when you perform a read operation on the SGMII/RGMII/SMII Status Register. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RO	0x0

gmacgrp_interrupt_mask

Register 15 (Interrupt Mask Register)

The Interrupt Mask Register bits enable you to mask the interrupt signal because of the corresponding event in the Interrupt Status Register.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF80003C
i_emac_emac1	0xFF802000	0xFF80203C
i_emac_emac2	0xFF804000	0xFF80403C

Offset: 0x3C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
reserved_31_11 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved_31_11 RO 0x0					lpiim RW 0x0	tsim RO 0x0	reserved_8_4 RO 0x0					pmtim RW 0x0	pcsancim RO 0x0	pcslchgim RO 0x0	rgsmiiim RO 0x0

gmacgrp_interrupt_mask Fields

Bit	Name	Description	Access	Reset
31:11	reserved_31_11	Reserved	RO	0x0

Bit	Name	Description	Access	Reset
10	lpiim	LPI Interrupt Mask When set, this bit disables the assertion of the interrupt signal because of the setting of the LPI Interrupt Status bit in Register 14 (Interrupt Status Register). **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
9	tsim	Timestamp Interrupt Mask When set, this bit disables the assertion of the interrupt signal because of the setting of Timestamp Interrupt Status bit in Register 14 (Interrupt Status Register). This bit is valid only when IEEE1588 timestamping is enabled. In all other modes, this bit is reserved. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RO	0x0
8:4	reserved_8_4	Reserved	RO	0x0
3	pmtim	PMT Interrupt Mask When set, this bit disables the assertion of the interrupt signal because of the setting of PMT Interrupt Status bit in Register 14 (Interrupt Status Register).	RW	0x0
2	pcsancim	PCS AN Completion Interrupt Mask When set, this bit disables the assertion of the interrupt signal because of the setting of PCS Auto-negotiation complete bit in Register 14 (Interrupt Status Register).	RO	0x0

Bit	Name	Description	Access	Reset
1	pcslchgim	PCS Link Status Interrupt Mask When set, this bit disables the assertion of the interrupt signal because of the setting of the PCS Link-status changed bit in Register 14 (Interrupt Status Register).	RO	0x0
0	rgsmiiim	RGMII or SMII Interrupt Mask When set, this bit disables the assertion of the interrupt signal because of the setting of the RGMII or SMII Interrupt Status bit in Register 14 (Interrupt Status Register). **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RO	0x0

gmacgrp_mac_address0_high

Register 16 (MAC Address0 High Register)

The MAC Address0 High register holds the upper 16 bits of the first 6-byte MAC address of the station. The first DA byte that is received on the (G)MII interface corresponds to the LS byte (Bits [7:0]) of the MAC Address Low register. For example, if 0x112233445566 is received (0x11 in lane 0 of the first column) on the (G)MII as the destination address, then the MacAddress0 Register [47:0] is compared with 0x665544332211.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address0 Low Register are written. For proper synchronization updates, the consecutive writes to this Address Low Register should be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800040
i_emac_emac1	0xFF802000	0xFF802040
i_emac_emac2	0xFF804000	0xFF804040

Offset: 0x40

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RO 0x1	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address0_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable This bit is always set to 1.	RO	0x1
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address0 [47:32] This field contains the upper 16 bits (47:32) of the first 6-byte MAC address. The MAC uses this field for filtering the received frames and inserting the MAC address in the Transmit Flow Control (PAUSE) Frames.	RW	0xFFFF

gmacgrp_mac_address0_low

Register 17 (MAC Address0 Low Register)

The MAC Address0 Low register holds the lower 32 bits of the first 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800044

Module Instance	Base Address	Register Address
i_emac_emac1	0xFF802000	0xFF802044
i_emac_emac2	0xFF804000	0xFF804044

Offset: 0x44

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address0_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address0 [31:0] This field contains the lower 32 bits of the first 6-byte MAC address. This is used by the MAC for filtering the received frames and inserting the MAC address in the Transmit Flow Control (PAUSE) Frames.	RW	0xFFFFFFFF

gmacgrp_mac_address1_high

Register 18 (MAC Address1 High Register)

The MAC Address1 High register holds the upper 16 bits of the second 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address1 Low Register are written. For proper synchronization updates, the consecutive writes to this Address Low Register should be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800048
i_emac_emac1	0xFF802000	0xFF802048
i_emac_emac2	0xFF804000	0xFF804048

Offset: 0x48

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address1_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the second MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
30	sa	Source Address When this bit is set, the MAC Address1[47:0] is used to compare with the SA fields of the received frame. When this bit is reset, the MAC Address1[47:0] is used to compare with the DA fields of the received frame. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
29	mbc_5	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address1 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0

Bit	Name	Description	Access	Reset
28	mbc_4	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address1 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0
27	mbc_3	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address1 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0

Bit	Name	Description	Access	Reset
26	mbc_2	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address1 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0
25	mbc_1	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address1 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0

Bit	Name	Description	Access	Reset
24	mbc_0	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address1 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). Value: Description 0x0: UNMASKED 0x1: MASKED	RW	0x0
23:16	reserved_23_16	Reserved	RO	0x0
15:0	addrhi	MAC Address1 [47:32] This field contains the upper 16 bits (47:32) of the second 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address1_low

Register 19 (MAC Address1 Low Register)

The MAC Address1 Low register holds the lower 32 bits of the second 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF80004C
i_emac_emac1	0xFF802000	0xFF80204C
i_emac_emac2	0xFF804000	0xFF80404C

Offset: 0x4C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address1_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address1 [31:0] This field contains the lower 32 bits of the second 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address2_high

Register 20 (MAC Address2 High Register)

The MAC Address2 High register holds the upper 16 bits of the third 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address2 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address2 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800050
i_emac_emac1	0xFF802000	0xFF802050
i_emac_emac2	0xFF804000	0xFF804050

Offset: 0x50

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address2_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the third MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
30	sa	Source Address When this bit is set, the MAC Address2[47:0] is used to compare with the SA fields of the received frame. When this bit is reset, the MAC Address2[47:0] is used to compare with the DA fields of the received frame. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
29	mbc_5	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address2 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0
28	mbc_4	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address2 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0

Bit	Name	Description	Access	Reset
27	mbc_3	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address2 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0
26	mbc_2	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address2 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0

Bit	Name	Description	Access	Reset
25	mbc_1	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address2 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0
24	mbc_0	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address2 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0
23:16	reserved_23_16	Reserved	RO	0x0
15:0	addrhi	MAC Address2 [47:32] This field contains the upper 16 bits (47:32) of the third 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address2_low

Register 21 (MAC Address2 Low Register)

The MAC Address2 Low register holds the lower 32 bits of the third 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800054
i_emac_emac1	0xFF802000	0xFF802054
i_emac_emac2	0xFF804000	0xFF804054

Offset: 0x54

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address2_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address2 [31:0] This field contains the lower 32 bits of the third 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address3_high

Register 22 (MAC Address3 High Register)

The MAC Address3 High register holds the upper 16 bits of the fourth 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address3 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address3 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800058
i_emac_emac1	0xFF802000	0xFF802058
i_emac_emac2	0xFF804000	0xFF804058

Offset: 0x58

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address3_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the fourth MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
30	sa	Source Address When this bit is set, the MAC Address3[47:0] is used to compare with the SA fields of the received frame. When this bit is reset, the MAC Address3[47:0] is used to compare with the DA fields of the received frame. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
29	mbc_5	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address3 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0
28	mbc_4	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address3 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0

Bit	Name	Description	Access	Reset
27	mbc_3	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address3 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0
26	mbc_2	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address3 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0

Bit	Name	Description	Access	Reset
25	mbc_1	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address3 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0
24	mbc_0	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address3 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0
23:16	reserved_23_16	Reserved	RO	0x0
15:0	addrhi	MAC Address3 [47:32] This field contains the upper 16 bits (47:32) of the fourth 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address3_low

Register 23 (MAC Address3 Low Register)

The MAC Address3 Low register holds the lower 32 bits of the fourth 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF80005C
i_emac_emac1	0xFF802000	0xFF80205C
i_emac_emac2	0xFF804000	0xFF80405C

Offset: 0x5C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address3_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address3 [31:0] This field contains the lower 32 bits of the fourth 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address4_high

Register 24 (MAC Address4 High Register)

The MAC Address4 High register holds the upper 16 bits of the fifth 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address4 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address4 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800060
i_emac_emac1	0xFF802000	0xFF802060
i_emac_emac2	0xFF804000	0xFF804060

Offset: 0x60

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address4_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the fifth MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
30	sa	Source Address When this bit is set, the MAC Address4[47:0] is used to compare with the SA fields of the received frame. When this bit is reset, the MAC Address4[47:0] is used to compare with the DA fields of the received frame. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
29	mbc_5	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address4 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0
28	mbc_4	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address4 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0

Bit	Name	Description	Access	Reset
27	mbc_3	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address4 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0
26	mbc_2	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address4 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0

Bit	Name	Description	Access	Reset
25	mbc_1	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address4 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0
24	mbc_0	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address4 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0
23:16	reserved_23_16	Reserved	RO	0x0
15:0	addrhi	MAC Address4 [47:32] This field contains the upper 16 bits (47:32) of the fifth 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address4_low

Register 25 (MAC Address4 Low Register)

The MAC Address4 Low register holds the lower 32 bits of the fifth 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800064
i_emac_emac1	0xFF802000	0xFF802064
i_emac_emac2	0xFF804000	0xFF804064

Offset: 0x64

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address4_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address4 [31:0] This field contains the lower 32 bits of the fifth 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address5_high

Register 26 (MAC Address5 High Register)

The MAC Address5 High register holds the upper 16 bits of the sixth 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address5 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address5 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800068
i_emac_emac1	0xFF802000	0xFF802068
i_emac_emac2	0xFF804000	0xFF804068

Offset: 0x68

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address5_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the sixth MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30	sa	Source Address When this bit is set, the MAC Address5[47:0] is used to compare with the SA fields of the received frame. When this bit is reset, the MAC Address5[47:0] is used to compare with the DA fields of the received frame. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
29	mbc_5	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address5 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0
28	mbc_4	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address5 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0

Bit	Name	Description	Access	Reset
27	mbc_3	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address5 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0
26	mbc_2	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address5 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0

Bit	Name	Description	Access	Reset
25	mbc_1	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address5 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0
24	mbc_0	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address5 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0
23:16	reserved_23_16	Reserved	RO	0x0
15:0	addrhi	MAC Address5 [47:32] This field contains the upper 16 bits (47:32) of the sixth 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address5_low

Register 27 (MAC Address5 Low Register)

The MAC Address5 Low register holds the lower 32 bits of the sixth 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF80006C
i_emac_emac1	0xFF802000	0xFF80206C
i_emac_emac2	0xFF804000	0xFF80406C

Offset: 0x6C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address5_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address5 [31:0] This field contains the lower 32 bits of the sixth 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address6_high

Register 28 (MAC Address6 High Register)

The MAC Address6 High register holds the upper 16 bits of the seventh 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address6 Low Register are written. For proper synchronization updates, the consecutive writes to this MAC Address6 Low Register should be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800070
i_emac_emac1	0xFF802000	0xFF802070
i_emac_emac2	0xFF804000	0xFF804070

Offset: 0x70

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address6_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the seventh MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
30	sa	Source Address When this bit is set, the MAC Address6[47:0] is used to compare with the SA fields of the received frame. When this bit is reset, the MAC Address6[47:0] is used to compare with the DA fields of the received frame. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
29	mbc_5	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address6 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0

Bit	Name	Description	Access	Reset
28	mbc_4	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address6 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0
27	mbc_3	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address6 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0

Bit	Name	Description	Access	Reset
26	mbc_2	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address6 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0
25	mbc_1	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address6 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0

Bit	Name	Description	Access	Reset
24	mbc_0	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address6 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). Value: Description 0x0: UNMASKED 0x1: MASKED	RW	0x0
23:16	reserved_23_16	Reserved	RO	0x0
15:0	addrhi	MAC Address6 [47:32] This field contains the upper 16 bits (47:32) of the seventh 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address6_low

Register 29 (MAC Address6 Low Register)

The MAC Address6 Low register holds the lower 32 bits of the seventh 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800074
i_emac_emac1	0xFF802000	0xFF802074
i_emac_emac2	0xFF804000	0xFF804074

Offset: 0x74

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address6_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address6 [31:0] This field contains the lower 32 bits of the seventh 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address7_high

Register 30 (MAC Address7 High Register)

The MAC Address7 High register holds the upper 16 bits of the eighth 6-byte MAC address of the station.
If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address7 Low Register are written. For proper synchronization updates, the consecutive writes to this MAC Address7 Low Register should be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800078
i_emac_emac1	0xFF802000	0xFF802078
i_emac_emac2	0xFF804000	0xFF804078

Offset: 0x78

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address7_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the eighth MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30	sa	Source Address When this bit is set, the MAC Address7[47:0] is used to compare with the SA fields of the received frame. When this bit is reset, the MAC Address7[47:0] is used to compare with the DA fields of the received frame. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
29	mbc_5	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address7 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0
28	mbc_4	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address7 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0

Bit	Name	Description	Access	Reset
27	mbc_3	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address7 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0
26	mbc_2	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address7 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0

Bit	Name	Description	Access	Reset
25	mbc_1	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address7 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0
24	mbc_0	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address7 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0
23:16	reserved_23_16	Reserved	RO	0x0
15:0	addrhi	MAC Address7 [47:32] This field contains the upper 16 bits (47:32) of the eighth 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address7_low

Register 31 (MAC Address7 Low Register)

The MAC Address7 Low register holds the lower 32 bits of the eighth 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF80007C
i_emac_emac1	0xFF802000	0xFF80207C
i_emac_emac2	0xFF804000	0xFF80407C

Offset: 0x7C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address7_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address7 [31:0] This field contains the lower 32 bits of the eighth 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address8_high

Register 32 (MAC Address8 High Register)

The MAC Address8 High register holds the upper 16 bits of the nineth 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address8 Low Register are written. For proper synchronization updates, the consecutive writes to this MAC Address8 Low Register should be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800080
i_emac_emac1	0xFF802000	0xFF802080
i_emac_emac2	0xFF804000	0xFF804080

Offset: 0x80

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address8_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the nineth MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
30	sa	Source Address When this bit is set, the MAC Address8[47:0] is used to compare with the SA fields of the received frame. When this bit is reset, the MAC Address8[47:0] is used to compare with the DA fields of the received frame. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
29	mbc_5	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address8 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0

Bit	Name	Description	Access	Reset
28	mbc_4	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address8 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0
27	mbc_3	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address8 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0

Bit	Name	Description	Access	Reset
26	mbc_2	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address8 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0
25	mbc_1	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address8 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0

Bit	Name	Description	Access	Reset
24	mbc_0	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address8 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). Value: Description 0x0: UNMASKED 0x1: MASKED	RW	0x0
23:16	reserved_23_16	Reserved	RO	0x0
15:0	addrhi	MAC Address8 [47:32] This field contains the upper 16 bits (47:32) of the nineth 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address8_low

Register 33 (MAC Address8 Low Register)

The MAC Address8 Low register holds the lower 32 bits of the nineth 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800084
i_emac_emac1	0xFF802000	0xFF802084
i_emac_emac2	0xFF804000	0xFF804084

Offset: 0x84

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address8_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address8 [31:0] This field contains the lower 32 bits of the nineth 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address9_high

Register 34 (MAC Address9 High Register)

The MAC Address9 High register holds the upper 16 bits of the tenth 6-byte MAC address of the station.
If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address9 Low Register are written. For proper synchronization updates, the consecutive writes to this MAC Address9 Low Register should be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800088
i_emac_emac1	0xFF802000	0xFF802088
i_emac_emac2	0xFF804000	0xFF804088

Offset: 0x88

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address9_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the tenth MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30	sa	Source Address When this bit is set, the MAC Address9[47:0] is used to compare with the SA fields of the received frame. When this bit is reset, the MAC Address9[47:0] is used to compare with the DA fields of the received frame. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
29	mbc_5	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address9 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0
28	mbc_4	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address9 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0

Bit	Name	Description	Access	Reset
27	mbc_3	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address9 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0
26	mbc_2	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address9 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0

Bit	Name	Description	Access	Reset
25	mbc_1	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address9 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0
24	mbc_0	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address9 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0
23:16	reserved_23_16	Reserved	RO	0x0
15:0	addrhi	MAC Address9 [47:32] This field contains the upper 16 bits (47:32) of the tenth 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address9_low

Register 35 (MAC Address9 Low Register)

The MAC Address9 Low register holds the lower 32 bits of the tenth 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF80008C
i_emac_emac1	0xFF802000	0xFF80208C
i_emac_emac2	0xFF804000	0xFF80408C

Offset: 0x8C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address9_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address9 [31:0] This field contains the lower 32 bits of the tenth 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address10_high

Register 36 (MAC Address10 High Register)

The MAC Address10 High register holds the upper 16 bits of the 11th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address10 Low Register are written. For proper synchronization updates, the consecutive writes to this MAC Address10 Low Register should be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800090
i_emac_emac1	0xFF802000	0xFF802090
i_emac_emac2	0xFF804000	0xFF804090

Offset: 0x90

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address10_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 11th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
30	sa	Source Address When this bit is set, the MAC Address10[47:0] is used to compare with the SA fields of the received frame. When this bit is reset, the MAC Address10[47:0] is used to compare with the DA fields of the received frame. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
29	mbc_5	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address10 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0

Bit	Name	Description	Access	Reset
28	mbc_4	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address10 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0
27	mbc_3	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address10 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0

Bit	Name	Description	Access	Reset
26	mbc_2	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address10 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0
25	mbc_1	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address10 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0

Bit	Name	Description	Access	Reset
24	mbc_0	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address10 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). Value: Description 0x0: UNMASKED 0x1: MASKED	RW	0x0
23:16	reserved_23_16	Reserved	RO	0x0
15:0	addrhi	MAC Address10 [47:32] This field contains the upper 16 bits (47:32) of the 11th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address10_low

Register 37 (MAC Address10 Low Register)

The MAC Address10 Low register holds the lower 32 bits of the 11th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800094
i_emac_emac1	0xFF802000	0xFF802094
i_emac_emac2	0xFF804000	0xFF804094

Offset: 0x94

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address10_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address10 [31:0] This field contains the lower 32 bits of the 11th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address11_high

Register 38 (MAC Address11 High Register)

The MAC Address11 High register holds the upper 16 bits of the 12th 6-byte MAC address of the station.
If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address11 Low Register are written. For proper synchronization updates, the consecutive writes to this MAC Address11 Low Register should be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800098
i_emac_emac1	0xFF802000	0xFF802098
i_emac_emac2	0xFF804000	0xFF804098

Offset: 0x98

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address11_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the twelfth MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30	sa	Source Address When this bit is set, the MAC Address11[47:0] is used to compare with the SA fields of the received frame. When this bit is reset, the MAC Address11[47:0] is used to compare with the DA fields of the received frame. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
29	mbc_5	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address11 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0
28	mbc_4	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address11 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0

Bit	Name	Description	Access	Reset
27	mbc_3	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address11 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0
26	mbc_2	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address11 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0

Bit	Name	Description	Access	Reset
25	mbc_1	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address11 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0
24	mbc_0	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address11 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0
23:16	reserved_23_16	Reserved	RO	0x0
15:0	addrhi	MAC Address11 [47:32] This field contains the upper 16 bits (47:32) of the 12th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address11_low

Register 39 (MAC Address1 Low Register)

The MAC Address11 Low register holds the lower 32 bits of the 12th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF80009C
i_emac_emac1	0xFF802000	0xFF80209C
i_emac_emac2	0xFF804000	0xFF80409C

Offset: 0x9C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address11_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address11 [31:0] This field contains the lower 32 bits of the 12th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address12_high

Register 40 (MAC Address12 High Register)

The MAC Address12 High register holds the upper 16 bits of the 13th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address13 Low Register are written. For proper synchronization updates, the consecutive writes to this MAC Address12 Low Register should be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8000A0
i_emac_emac1	0xFF802000	0xFF8020A0
i_emac_emac2	0xFF804000	0xFF8040A0

Offset: 0xA0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address12_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 13th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value: Description** 0x0: DISABLED 0x1: ENABLED	RW	0x0
30	sa	Source Address When this bit is set, the MAC Address12[47:0] is used to compare with the SA fields of the received frame. When this bit is reset, the MAC Address12[47:0] is used to compare with the DA fields of the received frame. **Value: Description** 0x0: DISABLED 0x1: ENABLED	RW	0x0
29	mbc_5	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address12 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value: Description** 0x0: UNMASKED 0x1: MASKED	RW	0x0

Bit	Name	Description	Access	Reset
28	mbc_4	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address12 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0
27	mbc_3	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address12 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0

Bit	Name	Description	Access	Reset
26	mbc_2	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address12 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0
25	mbc_1	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address12 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0

Bit	Name	Description	Access	Reset
24	mbc_0	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address12 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). Value: Description 0x0: UNMASKED 0x1: MASKED	RW	0x0
23:16	reserved_23_16	Reserved	RO	0x0
15:0	addrhi	MAC Address12 [47:32] This field contains the upper 16 bits (47:32) of the 13th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address12_low

Register 41 (MAC Address12 Low Register)

The MAC Address12 Low register holds the lower 32 bits of the 13th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8000A4
i_emac_emac1	0xFF802000	0xFF8020A4
i_emac_emac2	0xFF804000	0xFF8040A4

Offset: 0xA4

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address12_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address12 [31:0] This field contains the lower 32 bits of the 13th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address13_high

Register 42 (MAC Address13 High Register)

The MAC Address13 High register holds the upper 16 bits of the 14th 6-byte MAC address of the station.
If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address13 Low Register are written. For proper synchronization updates, the consecutive writes to this MAC Address13 Low Register should be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8000A8
i_emac_emac1	0xFF802000	0xFF8020A8
i_emac_emac2	0xFF804000	0xFF8040A8

Offset: 0xA8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address13_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 14th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
30	sa	Source Address When this bit is set, the MAC Address13[47:0] is used to compare with the SA fields of the received frame. When this bit is reset, the MAC Address13[47:0] is used to compare with the DA fields of the received frame. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
29	mbc_5	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address13 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0
28	mbc_4	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address13 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0

Bit	Name	Description	Access	Reset
27	mbc_3	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address13 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0
26	mbc_2	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address13 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0

Bit	Name	Description	Access	Reset
25	mbc_1	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address13 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0
24	mbc_0	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address13 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0
23:16	reserved_23_16	Reserved	RO	0x0
15:0	addrhi	MAC Address13 [47:32] This field contains the upper 16 bits (47:32) of the 14th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address13_low

Register 43 (MAC Address13 Low Register)

The MAC Address13 Low register holds the lower 32 bits of the 14th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8000AC
i_emac_emac1	0xFF802000	0xFF8020AC
i_emac_emac2	0xFF804000	0xFF8040AC

Offset: 0xAC

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address13_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address13 [31:0] This field contains the lower 32 bits of the 14th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address14_high

Register 44 (MAC Address14 High Register)

The MAC Address14 High register holds the upper 16 bits of the 15th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address15 Low Register are written. For proper synchronization updates, the consecutive writes to this MAC Address14 Low Register should be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8000B0
i_emac_emac1	0xFF802000	0xFF8020B0
i_emac_emac2	0xFF804000	0xFF8040B0

Offset: 0xB0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address14_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 15th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
30	sa	Source Address When this bit is set, the MAC Address14[47:0] is used to compare with the SA fields of the received frame. When this bit is reset, the MAC Address14[47:0] is used to compare with the DA fields of the received frame. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
29	mbc_5	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address14 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0

Bit	Name	Description	Access	Reset
28	mbc_4	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address14 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0
27	mbc_3	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address14 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0

Bit	Name	Description	Access	Reset
26	mbc_2	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address14 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0
25	mbc_1	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address14 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0

Bit	Name	Description	Access	Reset
24	mbc_0	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address14 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). Value: Description 0x0: UNMASKED 0x1: MASKED	RW	0x0
23:16	reserved_23_16	Reserved	RO	0x0
15:0	addrhi	MAC Address14 [47:32] This field contains the upper 16 bits (47:32) of the 15th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address14_low

Register 45 (MAC Address14 Low Register)

The MAC Address14 Low register holds the lower 32 bits of the 15th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8000B4
i_emac_emac1	0xFF802000	0xFF8020B4
i_emac_emac2	0xFF804000	0xFF8040B4

Offset: 0xB4

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address14_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address14 [31:0] This field contains the lower 32 bits of the 15th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address15_high

Register 46 (MAC Address15 High Register)

The MAC Address15 High register holds the upper 16 bits of the 16th 6-byte MAC address of the station.
If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address15 Low Register are written. For proper synchronization updates, the consecutive writes to this MAC Address15 Low Register should be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8000B8
i_emac_emac1	0xFF802000	0xFF8020B8
i_emac_emac2	0xFF804000	0xFF8040B8

Offset: 0xB8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address15_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 16th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30	sa	Source Address When this bit is set, the MAC Address15[47:0] is used to compare with the SA fields of the received frame. When this bit is reset, the MAC Address15[47:0] is used to compare with the DA fields of the received frame. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
29	mbc_5	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address15 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0
28	mbc_4	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address15 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0

Bit	Name	Description	Access	Reset
27	mbc_3	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address15 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0
26	mbc_2	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address15 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0

Bit	Name	Description	Access	Reset
25	mbc_1	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address15 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). Value: 0x0 — UNMASKED; 0x1 — MASKED	RW	0x0
24	mbc_0	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address15 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). Value: 0x0 — UNMASKED; 0x1 — MASKED	RW	0x0
23:16	reserved_23_16	Reserved	RO	0x0
15:0	addrhi	MAC Address15 [47:32] This field contains the upper 16 bits (47:32) of the 16th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address15_low

Register 47 (MAC Address15 Low Register)

The MAC Address15 Low register holds the lower 32 bits of the 16th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8000BC
i_emac_emac1	0xFF802000	0xFF8020BC
i_emac_emac2	0xFF804000	0xFF8040BC

Offset: 0xBC

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address15_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address15 [31:0] This field contains the lower 32 bits of the 16th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_sgmii_rgmii_smii_control_status

The SGMII/RGMII/SMII Status register indicates the status signals received by the RGMII interface (selected at reset) from the PHY.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8000D8

Module Instance	Base Address	Register Address
i_emac_emac1	0xFF802000	0xFF8020D8
i_emac_emac2	0xFF804000	0xFF8040D8

Offset: 0xD8

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												lnksts RO 0x0	lnkspeed RO 0x0		lnkmod RO 0x0

gmacgrp_sgmii_rgmii_smii_control_status Fields

Bit	Name	Description	Access	Reset
3	lnksts	This bit indicates whether the link is up (1'b1) or down (1'b0). Value: Description 0x0: LINKDOWN 0x1: LINKUP	RO	0x0
2:1	lnkspeed	This bit indicates the current speed of the link. Bit 2 is reserved when the MAC is configured for the SMII PHY interface. Value: Description 0x0: SPEED2POINT5MHZ 0x1: SPEED25MHZ 0x2: SPEED125MHZ	RO	0x0

Bit	Name	Description	Access	Reset
0	lnkmod	This bit indicates the current mode of operation of the link Value: Description 0x0: HALFDUP 0x1: FULLDUP	RO	0x0

gmacgrp_wdog_timeout

Register 55 (Watchdog Timeout Register)
This register controls the watchdog timeout for received frames.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8000DC
i_emac_emac1	0xFF802000	0xFF8020DC
i_emac_emac2	0xFF804000	0xFF8040DC

Offset: 0xDC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
reserved_31_17 RO 0x0															pwe RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved_15_14 RO 0x0		wto RW 0x0													

gmacgrp_wdog_timeout Fields

Bit	Name	Description	Access	Reset
31:17	reserved_31_17	Reserved	RO	0x0

Bit	Name	Description	Access	Reset
16	pwe	Programmable Watchdog Enable When this bit is set and Bit 23 (WD) of Register 0 (MAC Configuration Register) is reset, the WTO field (Bits[13:0]) is used as watchdog timeout for a received frame. When this bit is cleared, the watchdog timeout for a received frame is controlled by the setting of Bit 23 (WD) and Bit 20 (JE) in Register 0 (MAC Configuration Register).	RW	0x0
15:14	reserved_15_14	Reserved	RO	0x0
13:0	wto	Watchdog Timeout When Bit 16 (PWE) is set and Bit 23 (WD) of Register 0 (MAC Configuration Register) is reset, this field is used as watchdog timeout for a received frame. If the length of a received frame exceeds the value of this field, such frame is terminated and declared as an error frame. Note: When Bit 16 (PWE) is set, the value in this field should be more than 1,522 (0x05F2). Otherwise, the IEEE Std 802.3-specified valid tagged frames are declared as error frames and are dropped.	RW	0x0

gmacgrp_genpio

Register 56 (General Purpose IO Register)
This register provides the control to drive up to 4 bits of output ports (GPO) and the status of up to 4 input ports (GPIS). It also provides the control to generate interrupts on events occurring on the gpi_i pin.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8000E0
i_emac_emac1	0xFF802000	0xFF8020E0
i_emac_emac2	0xFF804000	0xFF8040E0

Offset: 0xE0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
reserved_31_x RO 0x0							gpit RW 0x0	reserved_23_x RO 0x0							gpie RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved_15_x RO 0x0							gpo RW 0x0	reserved_7_x RO 0x0							gpis RO 0x0

gmacgrp_genpio Fields

Bit	Name	Description	Access	Reset
31:25	reserved_31_x	Reserved	RO	0x0
24	gpit	GPI Type When set, this bit indicates that the corresponding GPIS is of latched-low (LL) type. When reset, this bit indicates that the corresponding GPIS is of latched-high (LH) type. The number of bits available in this field depend on the GP Input Signal Width option. Other bits are not used (reserved and always reset).	RW	0x0
23:17	reserved_23_x	Reserved	RO	0x0

Bit	Name	Description	Access	Reset
16	gpie	GPI Interrupt Enable When this bit is set and the programmed event (LL or LH) occurs on the corresponding GPIS bit, Bit 11 (GPIIS) of Register 14 (Interrupt Status Register) is set. Accordingly, the interrupt is generated on the mci_intr_o or sbd_intr_o. The GPIIS bit is cleared when the host reads the Bits[7:0] of this register. When reset, Bit 11 (GPIIS) of Register 14 (Interrupt Status Register) is not set when any event occurs on the corresponding GPIS bits. The number of bits available in this field depend on the GP Input Signal Width option. Other bits are not used (reserved and always reset).	RW	0x0
15:9	reserved_15_x	Reserved	RO	0x0
8	gpo	General Purpose Output When this bit is set, it directly drives the gpo_o output ports. When this bit is reset, it does not directly drive the gpo_o output ports. The number of bits available in this field depend on the GP Output Signal Width option. Other bits are not used (reserved and always reset).	RW	0x0
7:1	reserved_7_x	Reserved	RO	0x0

Bit	Name	Description	Access	Reset
0	gpis	General Purpose Input Status This field gives the status of the signals connected to the gpi_i input ports. This field is of the following types based on the setting of the corresponding GPIT field of this register: * Latched-low (LL): This field is cleared when the corresponding gpi_i input becomes low. This field remains low until the host reads this field. After this, this field reflects the current value of the gpi_i input. * Latched-high (LH) : This field is set when the corresponding gpi_i input becomes high. This field remains high until the host reads this field. After this, this field reflects the current value of the gpi_i input. The number of bits available in this field depend on the GP Input Signal Width option. Other bits are not used (reserved and always reset).	RO	0x0

gmacgrp_mmc_control

Register 64 (MMC Control Register)

The MMC Control register establishes the operating mode of the management counters.

Note:
The bit 0 (Counters Reset) has higher priority than bit 4 (Counter Preset). Therefore, when the Software tries to set both bits in the same write cycle, all counters are cleared and the bit 4 is not set.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800100
i_emac_emac1	0xFF802000	0xFF802100

Module Instance	Base Address	Register Address
i_emac_emac2	0xFF804000	0xFF804100

Offset: 0x100

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
reserved_31_9 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved_31_9 RO 0x0							ucdbc RW 0x0	reserved_7_6 RO 0x0		cntprstlvl RW 0x0	cntprst RW 0x0	cntfreez RW 0x0	rstonrd RW 0x0	cntstopro RW 0x0	cntrst RW 0x0

gmacgrp_mmc_control Fields

Bit	Name	Description	Access	Reset
31:9	reserved_31_9	Reserved	RO	0x0
8	ucdbc	Update MMC Counters for Dropped Broadcast Frames When set, this bit enables MAC to update all the related MMC Counters for Broadcast frames dropped due to setting of DBF bit (Disable Broadcast Frames) of MAC Filter Register at offset 0x0004. When reset, MMC Counters are not updated for dropped Broadcast frames.	RW	0x0
7:6	reserved_7_6	Reserved	RO	0x0

Bit	Name	Description	Access	Reset
5	cntprstlvl	Full-Half Preset When low and bit 4 is set, all MMC counters get preset to almost-half value. All octet counters get preset to 0x7FFF_F800 (half - 2KBytes) and all frame-counters gets preset to 0x7FFF_FFF0 (half - 16). When this bit is high and bit 4 is set, all MMC counters get preset to almost-full value. All octet counters get preset to 0xFFFF_F800 (full - 2KBytes) and all frame-counters gets preset to 0xFFFF_FFF0 (full - 16). For 16-bit counters, the almost-half preset values are 0x7800 and 0x7FF0 for the respective octet and frame counters. Similarly, the almost-full preset values for the 16-bit counters are 0xF800 and 0xFFF0. **Value** **Description** 0x0 ALMOSTHALF 0x1 ALMOSTFULL	RW	0x0
4	cntprst	Counters Preset When this bit is set, all counters are initialized or preset to almost full or almost half according to bit 5. This bit is cleared automatically after 1 clock cycle. This bit, along with bit 5, is useful for debugging and testing the assertion of interrupts because of MMC counter becoming half-full or full. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
3	cntfreez	MMC Counter Freeze When this bit is set, it freezes all MMC counters to their current value. Until this bit is reset to 0, no MMC counter is updated because of any transmitted or received frame. If any MMC counter is read with the Reset on Read bit set, then that counter is also cleared in this mode. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
2	rstonrd	Reset on Read When this bit is set, the MMC counters are reset to zero after Read (self-clearing after reset). The counters are cleared when the least significant byte lane (bits[7:0]) is read. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
1	cntstopro	Counters Stop Rollover When this bit is set, after reaching maximum value, the counter does not roll over to zero. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
0	cntrst	Counters Reset When this bit is set, all counters are reset. This bit is cleared automatically after one clock cycle. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0

gmacgrp_mmc_receive_interrupt

Register 65 (MMC Receive Interrupt Register)

The MMC Receive Interrupt register maintains the interrupts that are generated when the following happens:
 * Receive statistic counters reach half of their maximum values (0x8000_0000 for 32-bit counter and 0x8000 for 16-bit counter).
 * Receive statistic counters cross their maximum values (0xFFFF_FFFF for 32-bit counter and 0xFFFF for 16-bit counter).
When the Counter Stop Rollover is set, then interrupts are set but the counter remains at all-ones. The MMC Receive Interrupt register is a 32-bit wide register. An interrupt bit is cleared when the respective MMC counter that caused the interrupt is read. The least significant byte lane (Bits[7:0]) of the respective counter must be read in order to clear the interrupt bit.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800104
i_emac_emac1	0xFF802000	0xFF802104
i_emac_emac2	0xFF804000	0xFF804104

Offset: 0x104

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
reserved_31_26 RO 0x0						rxctrlfis RO 0x0	rxrcverrfis RO 0x0	rxwdogfis RO 0x0	rxvlangbfis RO 0x0	rxfovfis RO 0x0	rxpausfis RO 0x0	rxorangefis RO 0x0	rxlenerfis RO 0x0	rxucgfis RO 0x0	rx1024tmaxoctgbfis RO 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rx512t1023octgbfis RO 0x0	rx256t511octgbfis RO 0x0	rx128t255octgbfis RO 0x0	rx65t127octgbfis RO 0x0	rx64octgbfis RO 0x0	rxosizegfis RO 0x0	rxusizegfis RO 0x0	rxjaberfis RO 0x0	rxruntfis RO 0x0	rxalgnerfis RO 0x0	rxcrcerfis RO 0x0	rxmcgfis RO 0x0	rxbcgfis RO 0x0	rxgoctis RO 0x0	rxgboctis RO 0x0	rxgbfrmis RO 0x0

gmacgrp_mmc_receive_interrupt Fields

Bit	Name	Description	Access	Reset
31:26	reserved_31_26	Reserved	RO	0x0
25	rxctrlfis	MMC Receive Control Frame Counter Interrupt Status This bit is set when the rxctrlframes_g counter reaches half of the maximum value or the maximum value.	RO	0x0
24	rxrcverrfis	MMC Receive Error Frame Counter Interrupt Status This bit is set when the rxrcverror counter reaches half of the maximum value or the maximum value.	RO	0x0
23	rxwdogfis	MMC Receive Watchdog Error Frame Counter Interrupt Status This bit is set when the rxwatchdo-gerror counter reaches half of the maximum value or the maximum value. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RO	0x0

Bit	Name	Description	Access	Reset
22	rxvlangbfis	MMC Receive VLAN Good Bad Frame Counter Interrupt Status This bit is set when the rxvlanframes_gb counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0
21	rxfovfis	MMC Receive FIFO Overflow Frame Counter Interrupt Status This bit is set when the rxfifoover-flow counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0
20	rxpausfis	MMC Receive Pause Frame Counter Interrupt Status This bit is set when the rxpause-frame counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0
19	rxorangefis	MMC Receive Out Of Range Error Frame Counter Interrupt Status This bit is set when the rxoutofran-getype counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0

Bit	Name	Description	Access	Reset
18	rxlenerfis	MMC Receive Length Error Frame Counter Interrupt Status This bit is set when the rxlength-error counter reaches half of the maximum value or the maximum value. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RO	0x0
17	rxucgfis	MMC Receive Unicast Good Frame Counter Interrupt Status This bit is set when the rxunicast-frames_gb counter reaches half of the maximum value or the maximum value. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RO	0x0
16	rx1024tmaxoctgbfis	MMC Receive 1024 to Maximum Octet Good Bad Frame Counter Interrupt Status This bit is set when the rx1024tomaxoctets_gb counter reaches half of the maximum value or the maximum value. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RO	0x0

Bit	Name	Description	Access	Reset
15	rx512t1023octgbfis	MMC Receive 512 to 1023 Octet Good Bad Frame Counter Interrupt Status This bit is set when the rx512to1023octets_gb counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0
14	rx256t511octgbfis	MMC Receive 256 to 511 Octet Good Bad Frame Counter Interrupt Status This bit is set when the rx256to511octets_gb counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0
13	rx128t255octgbfis	MMC Receive 128 to 255 Octet Good Bad Frame Counter Interrupt Status This bit is set when the rx128to255octets_gb counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0

Bit	Name	Description	Access	Reset
12	rx65t127octgbfis	MMC Receive 65 to 127 Octet Good Bad Frame Counter Interrupt Status This is set when the rx65to127octets_gb counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0
11	rx64octgbfis	MMC Receive 64 Octet Good Bad Frame Counter Interrupt Status This bit is set when the rx64octets_gb counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0
10	rxosizegfis	MMC Receive Oversize Good Frame Counter Interrupt Status This bit is set when the rxoversize_g counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0
9	rxusizegfis	MMC Receive Undersize Good Frame Counter Interrupt Status This bit is set when the rxunder-size_g counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0

Bit	Name	Description	Access	Reset
8	rxjaberfis	MMC Receive Jabber Error Frame Counter Interrupt Status This bit is set when the rxjabber-error counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0
7	rxruntfis	MMC Receive Runt Frame Counter Interrupt Status This bit is set when the rxrunterror counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0
6	rxalgnerfis	MMC Receive Alignment Error Frame Counter Interrupt Status This bit is set when the rxalignmen-terror counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0
5	rxcrcerfis	MMC Receive CRC Error Frame Counter Interrupt Status This bit is set when the rxcrcerror counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0

Bit	Name	Description	Access	Reset
4	rxmcgfis	MMC Receive Multicast Good Frame Counter Interrupt Status This bit is set when the rxmulticastframes_g counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0
3	rxbcgfis	MMC Receive Broadcast Good Frame Counter Interrupt Status. This bit is set when the rxbroadcastframes_g counter reaches half of the maximum value or the maximum value.	RO	0x0
2	rxgoctis	MMC Receive Good Octet Counter Interrupt Status. This bit is set when the rxoctetcount_g counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0
1	rxgboctis	MMC Receive Good Bad Octet Counter Interrupt Status This bit is set when the rxoctetcount_bg counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0

Bit	Name	Description	Access	Reset
0	rxgbfrmis	MMC Receive Good Bad Frame Counter Interrupt Status This bit is set when the rxframe-count_bg counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 ALMOSTHALF 0x1 ALMOSTFULL	RO	0x0

gmacgrp_mmc_transmit_interrupt

Register 66 (MMC Transmit Interrupt Register)

The MMC Transmit Interrupt register maintains the interrupts generated when transmit statistic counters reach half of their maximum values (0x8000_0000 for 32-bit counter and 0x8000 for 16-bit counter), and the maximum values (0xFFFF_FFFF for 32-bit counter and 0xFFFF for 16-bit counter). When Counter Stop Rollover is set, then interrupts are set but the counter remains at all-ones. The MMC Transmit Interrupt register is a 32-bit wide register. An interrupt bit is cleared when the respective MMC counter that caused the interrupt is read. The least significant byte lane (Bits[7:0]) of the respective counter must be read in order to clear the interrupt bit.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800108
i_emac_emac1	0xFF802000	0xFF802108
i_emac_emac2	0xFF804000	0xFF804108

Offset: 0x108

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
reserved_31_26 RO 0x0						txosizegfis RO 0x0	txvlangfis RO 0x0	txpausfis RO 0x0	txexdeffis RO 0x0	txgfrmis RO 0x0	txgoctis RO 0x0	txcarerfis RO 0x0	txexcolfis RO 0x0	txlatcolfis RO 0x0	txdeffis RO 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
txmcolgfis RO 0x0	txscolgfis RO 0x0	txuflowerfis RO 0x0	txbcgbfis RO 0x0	txmcgbfis RO 0x0	txucgbfis RO 0x0	tx1024tmaxoctgbfis RO 0x0	tx512t1023octgbfis RO 0x0	tx256t511octgbfis RO 0x0	tx128t255octgbfis RO 0x0	tx65t127octgbfis RO 0x0	tx64octgbfis RO 0x0	txmcgfis RO 0x0	txbcgfis RO 0x0	txgbfrmis RO 0x0	txgboctis RO 0x0

gmacgrp_mmc_transmit_interrupt Fields

Bit	Name	Description	Access	Reset
31:26	reserved_31_26	Reserved	RO	0x0
25	txosizegfis	MMC Transmit Oversize Good Frame Counter Interrupt Status This bit is set when the txoversize_g counter reaches half of the maximum value or the maximum value.	RO	0x0
24	txvlangfis	MMC Transmit VLAN Good Frame Counter Interrupt Status This bit is set when the txvlanframes_g counter reaches half of the maximum value or the maximum value. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RO	0x0

Bit	Name	Description	Access	Reset
23	txpausfis	MMC Transmit Pause Frame Counter Interrupt Status This bit is set when the txpausefra-meserror counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RO	0x0
22	txexdeffis	MMC Transmit Excessive Deferral Frame Counter Interrupt Status This bit is set when the txexcessdef counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RO	0x0
21	txgfrmis	MMC Transmit Good Frame Counter Interrupt Status This bit is set when the txframe-count_g counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RO	0x0
20	txgoctis	MMC Transmit Good Octet Counter Interrupt Status This bit is set when the txoctet-count_g counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RO	0x0

Bit	Name	Description	Access	Reset
19	txcarerfis	MMC Transmit Carrier Error Frame Counter Interrupt Status This bit is set when the txcarrier-error counter reaches half of the maximum value or the maximum value. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RO	0x0
18	txexcolfis	MMC Transmit Excessive Collision Frame Counter Interrupt Status This bit is set when the txexcesscol counter reaches half of the maximum value or the maximum value. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RO	0x0
17	txlatcolfis	MMC Transmit Late Collision Frame Counter Interrupt Status This bit is set when the txlatecol counter reaches half of the maximum value or the maximum value. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RO	0x0
16	txdeffis	MMC Transmit Deferred Frame Counter Interrupt Status This bit is set when the txdeferred counter reaches half of the maximum value or the maximum value. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RO	0x0

Bit	Name	Description	Access	Reset
15	txmcolgfis	MMC Transmit Multiple Collision Good Frame Counter Interrupt Status This bit is set when the txmulticol_g counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RO	0x0
14	txscolgfis	MMC Transmit Single Collision Good Frame Counter Interrupt Status This bit is set when the txsinglecol_g counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RO	0x0
13	txuflowerfis	MMC Transmit Underflow Error Frame Counter Interrupt Status This bit is set when the txunderflowerror counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RO	0x0
12	txbcgbfis	MMC Transmit Broadcast Good Bad Frame Counter Interrupt Status This bit is set when the txbroadcastframes_gb counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RO	0x0

Bit	Name	Description	Access	Reset
11	txmcgbfis	MMC Transmit Multicast Good Bad Frame Counter Interrupt Status This bit is set when the txmulticastframes_gb counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0
10	txucgbfis	MMC Transmit Unicast Good Bad Frame Counter Interrupt Status This bit is set when the txunicastframes_gb counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0
9	tx1024tmaxoctgbfis	MMC Transmit 1024 to Maximum Octet Good Bad Frame Counter Interrupt Status This bit is set when the tx1024tomaxoctets_gb counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0

Bit	Name	Description	Access	Reset
8	tx512t1023octgbfis	MMC Transmit 512 to 1023 Octet Good Bad Frame Counter Interrupt Status This bit is set when the tx512to1023octets_gb counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0
7	tx256t511octgbfis	MMC Transmit 256 to 511 Octet Good Bad Frame Counter Interrupt Status This bit is set when the tx256to511octets_gb counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0
6	tx128t255octgbfis	MMC Transmit 128 to 255 Octet Good Bad Frame Counter Interrupt Status This bit is set when the tx128to255octets_gb counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0

Bit	Name	Description	Access	Reset
5	tx65t127octgbfis	MMC Transmit 65 to 127 Octet Good Bad Frame Counter Interrupt Status This bit is set when the tx65to127octets_gb counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0
4	tx64octgbfis	MMC Transmit 64 Octet Good Bad Frame Counter Interrupt Status. This bit is set when the tx64octets_gb counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RO	0x0
3	txmcgfis	MMC Transmit Multicast Good Frame Counter Interrupt Status This bit is set when the txmulti-castframes_g counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0

Bit	Name	Description	Access	Reset
2	txbcgfis	MMC Transmit Broadcast Good Frame Counter Interrupt Status This bit is set when the txbroadcastframes_g counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0
1	txgbfrmis	MMC Transmit Good Bad Frame Counter Interrupt Status This bit is set when the txframecount_gb counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0
0	txgboctis	MMC Transmit Good Bad Octet Counter Interrupt Status This bit is set when the txoctetcount_gb counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0

gmacgrp_mmc_receive_interrupt_mask

Regsiter 67 (MMC Receive Interrupt Mask Register)

The MMC Receive Interrupt Mask register maintains the masks for the interrupts generated when the receive statistic counters reach half of their maximum value, or maximum value. This register is 32-bits wide.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF80010C
i_emac_emac1	0xFF802000	0xFF80210C
i_emac_emac2	0xFF804000	0xFF80410C

Offset: 0x10C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
reserved_31_26 RO 0x0						rxctrlfim RW 0x0	rxrcverrfim RW 0x0	rxwdogfim RW 0x0	rxvlangbfim RW 0x0	rxfovfim RW 0x0	rxpausfim RW 0x0	rxorangefim RW 0x0	rxlenerfim RW 0x0	rxucgfim RW 0x0	rx1024tmaxoctgbfim RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rx512t1023octgbfim RW 0x0	rx256t511octgbfim RW 0x0	rx128t255octgbfim RW 0x0	rx65t127octgbfim RW 0x0	rx64octgbfim RW 0x0	rxosizegfim RW 0x0	rxusizegfim RW 0x0	rxjaberfim RW 0x0	rxruntfim RW 0x0	rxalgnerfim RW 0x0	rxcrcerfim RW 0x0	rxmcgfim RW 0x0	rxbcgfim RW 0x0	rxgoctim RW 0x0	rxgboctim RW 0x0	rxgbfrmim RW 0x0

gmacgrp_mmc_receive_interrupt_mask Fields

Bit	Name	Description	Access	Reset
31:26	reserved_31_26	Reserved	RO	0x0
25	rxctrlfim	MMC Receive Control Frame Counter Interrupt Mask Setting this bit masks the interrupt when the rxctrlframes counter reaches half the maximum value, and also when it reaches the maximum value. **Value** — **Description** 0x0 — NOMASKINTR 0x1 — MASKINTR	RW	0x0

Bit	Name	Description	Access	Reset
24	rxrcverrfim	MMC Receive Error Frame Counter Interrupt Mask Setting this bit masks the interrupt when the rxrcverror error counter reaches half the maximum value, and also when it reaches the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0
23	rxwdogfim	MMC Receive Watchdog Error Frame Counter Interrupt Mask Setting this bit masks the interrupt when the rxwatchdog counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0
22	rxvlangbfim	MMC Receive VLAN Good Bad Frame Counter Interrupt Mask Setting this bit masks the interrupt when the rxvlanframes_gb counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0

Bit	Name	Description	Access	Reset
21	rxfovfim	MMC Receive FIFO Overflow Frame Counter Interrupt Mask Setting this bit masks the interrupt when the rxfifooverflow counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0
20	rxpausfim	MMC Receive Pause Frame Counter Interrupt Mask Setting this bit masks the interrupt when the rxpauseframes counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0
19	rxorangefim	MMC Receive Out Of Range Error Frame Counter Interrupt Mask Setting this bit masks the interrupt when the rxoutofrangetype counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0

Bit	Name	Description	Access	Reset
18	rxlenerfim	MMC Receive Length Error Frame Counter Interrupt Mask Setting this bit masks the interrupt when the rxlengtherror counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0
17	rxucgfim	MMC Receive Unicast Good Frame Counter Interrupt Mask Setting this bit masks the interrupt when the rxunicastframes_g counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0
16	rx1024tmaxoctgbfim	MMC Receive 1024 to Maximum Octet Good Bad Frame Counter Interrupt Mask Setting this bit masks the interrupt when the rx1024tomaxoc-tets_gb counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0

Bit	Name	Description	Access	Reset
15	rx512t1023octgbfim	MMC Receive 512 to 1023 Octet Good Bad Frame Counter Interrupt Mask Setting this bit masks the interrupt when the rx512to1023octets_gb counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0
14	rx256t511octgbfim	MMC Receive 256 to 511 Octet Good Bad Frame Counter Interrupt Mask Setting this bit masks the interrupt when the rx256to511octets_gb counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0
13	rx128t255octgbfim	MMC Receive 128 to 255 Octet Good Bad Frame Counter Interrupt Mask Setting this bit masks the interrupt when the rx128to255octets_gb counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0

Bit	Name	Description	Access	Reset
12	rx65t127octgbfim	MMC Receive 65 to 127 Octet Good Bad Frame Counter Interrupt Mask Setting this bit masks the interrupt when the rx65to127octets_gb counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0
11	rx64octgbfim	MMC Receive 64 Octet Good Bad Frame Counter Interrupt Mask Setting this bit masks the interrupt when the rx64octets_gb counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0
10	rxosizegfim	MMC Receive Oversize Good Frame Counter Interrupt Mask Setting this bit masks the interrupt when the rxoversize_g counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0

Bit	Name	Description	Access	Reset
9	rxusizegfim	MMC Receive Undersize Good Frame Counter Interrupt Mask Setting this bit masks the interrupt when the rxundersize_g counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0
8	rxjaberfim	MMC Receive Jabber Error Frame Counter Interrupt Mask Setting this bit masks the interrupt when the rxjabbererror counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0
7	rxruntfim	MMC Receive Runt Frame Counter Interrupt Mask Setting this bit masks the interrupt when the rxrunterror counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0

Bit	Name	Description	Access	Reset
6	rxalgnerfim	MMC Receive Alignment Error Frame Counter Interrupt Mask Setting this bit masks the interrupt when the rxalignmenterror counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0
5	rxcrcerfim	MMC Receive CRC Error Frame Counter Interrupt Mask Setting this bit masks the interrupt when the rxcrcerror counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0
4	rxmcgfim	MMC Receive Multicast Good Frame Counter Interrupt Mask Setting this bit masks the interrupt when the rxmulticast-frames_g counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0

Bit	Name	Description	Access	Reset
3	rxbcgfim	MMC Receive Broadcast Good Frame Counter Interrupt Mask Setting this bit masks the interrupt when the rxbroadcast-frames_g counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0
2	rxgoctim	MMC Receive Good Octet Counter Interrupt Mask Setting this bit masks the interrupt when the rxoctetcount_g counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0
1	rxgboctim	MMC Receive Good Bad Octet Counter Interrupt Mask Setting this bit masks the interrupt when the rxoctetcount_gb counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0

Bit	Name	Description	Access	Reset
0	rxgbfrmim	MMC Receive Good Bad Frame Counter Interrupt Mask Setting this bit masks the interrupt when the rxframecount_gb counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0

gmacgrp_mmc_transmit_interrupt_mask

Register 68 (MMC Transmit Interrupt Mask Register)

The MMC Transmit Interrupt Mask register maintains the masks for the interrupts generated when the transmit statistic counters reach half of their maximum value or maximum value. This register is 32-bits wide.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800110
i_emac_emac1	0xFF802000	0xFF802110
i_emac_emac2	0xFF804000	0xFF804110

Offset: 0x110

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
reserved_31_26 RO 0x0						txosizegfim RW 0x0	txvlangfim RW 0x0	txpausfim RW 0x0	txexdeffim RW 0x0	txgfrmim RW 0x0	txgoctim RW 0x0	txcarerfim RW 0x0	txexcolfim RW 0x0	txlatcolfim RW 0x0	txdeffim RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
txmcolgfim RW 0x0	txscolgfim RW 0x0	txuflowerfim RW 0x0	txbcgbfim RW 0x0	txmcgbfim RW 0x0	txucgbfim RW 0x0	tx1024tmaxoctgbfim RW 0x0	tx512t1023octgbfim RW 0x0	tx256t511octgbfim RW 0x0	tx128t255octgbfim RW 0x0	tx65t127octgbfim RW 0x0	tx64octgbfim RW 0x0	txmcgfim RW 0x0	txbcgfim RW 0x0	txgbfrmim RW 0x0	txgboctim RW 0x0

gmacgrp_mmc_transmit_interrupt_mask Fields

Bit	Name	Description	Access	Reset
31:26	reserved_31_26	Reserved	RO	0x0
25	txosizegfim	MMC Transmit Oversize Good Frame Counter Interrupt Mask Setting this bit masks the interrupt when the txoversize_g counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0
24	txvlangfim	MMC Transmit VLAN Good Frame Counter Interrupt Mask Setting this bit masks the interrupt when the txvlanframes_g counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0

Bit	Name	Description	Access	Reset
23	txpausfim	MMC Transmit Pause Frame Counter Interrupt Mask Setting this bit masks the interrupt when the txpauseframes counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0
22	txexdeffim	MMC Transmit Excessive Deferral Frame Counter Interrupt Mask Setting this bit masks the interrupt when the txexcessdef counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0
21	txgfrmim	MMC Transmit Good Frame Counter Interrupt Mask Setting this bit masks the interrupt when the txframecount_g counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0

Bit	Name	Description	Access	Reset
20	txgoctim	MMC Transmit Good Octet Counter Interrupt Mask Setting this bit masks the interrupt when the txoctetcount_g counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0
19	txcarerfim	MMC Transmit Carrier Error Frame Counter Interrupt Mask Setting this bit masks the interrupt when the txcarriererror counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0
18	txexcolfim	MMC Transmit Excessive Collision Frame Counter Interrupt Mask Setting this bit masks the interrupt when the txexcesscol counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0

Bit	Name	Description	Access	Reset
17	txlatcolfim	MMC Transmit Late Collision Frame Counter Interrupt Mask Setting this bit masks the interrupt when the txlatecol counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0
16	txdeffim	MMC Transmit Deferred Frame Counter Interrupt Mask Setting this bit masks the interrupt when the txdeferred counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0
15	txmcolgfim	MMC Transmit Multiple Collision Good Frame Counter Interrupt Mask Setting this bit masks the interrupt when the txmulticol_g counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0

Bit	Name	Description	Access	Reset
14	txscolgfim	MMC Transmit Single Collision Good Frame Counter Interrupt Mask Setting this bit masks the interrupt when the txsinglecol_g counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0
13	txuflowerfim	MMC Transmit Underflow Error Frame Counter Interrupt Mask Setting this bit masks the interrupt when the txunderflowerror counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0
12	txbcgbfim	MMC Transmit Broadcast Good Bad Frame Counter Interrupt Mask Setting this bit masks the interrupt when the txbroadcast-frames_gb counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0

Bit	Name	Description	Access	Reset
11	txmcgbfim	MMC Transmit Multicast Good Bad Frame Counter Interrupt Mask Setting this bit masks the interrupt when the txmulticast-frames_gb counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0
10	txucgbfim	MMC Transmit Unicast Good Bad Frame Counter Interrupt Mask Setting this bit masks the interrupt when the txunicastframes_gb counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0
9	tx1024tmaxoctgbfim	MMC Transmit 1024 to Maximum Octet Good Bad Frame Counter Interrupt Mask Setting this bit masks the interrupt when the tx1024tomaxoc-tets_gb counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0

Bit	Name	Description	Access	Reset
8	tx512t1023octgbfim	MMC Transmit 512 to 1023 Octet Good Bad Frame Counter Interrupt Mask Setting this bit masks the interrupt when the tx512to1023octets_gb counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0
7	tx256t511octgbfim	MMC Transmit 256 to 511 Octet Good Bad Frame Counter Interrupt Mask Setting this bit masks the interrupt when the tx256to511octets_gb counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0
6	tx128t255octgbfim	MMC Transmit 128 to 255 Octet Good Bad Frame Counter Interrupt Mask Setting this bit masks the interrupt when the tx128to255octets_gb counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0

Bit	Name	Description	Access	Reset
5	tx65t127octgbfim	MMC Transmit 65 to 127 Octet Good Bad Frame Counter Interrupt Mask Setting this bit masks the interrupt when the tx65to127octets_gb counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0
4	tx64octgbfim	MMC Transmit 64 Octet Good Bad Frame Counter Interrupt Mask Setting this bit masks the interrupt when the tx64octets_gb counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0
3	txmcgfim	MMC Transmit Multicast Good Frame Counter Interrupt Mask Setting this bit masks the interrupt when the txmulticast-frames_g counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0

Bit	Name	Description	Access	Reset
2	txbcgfim	MMC Transmit Broadcast Good Frame Counter Interrupt Mask Setting this bit masks the interrupt when the txbroadcast-frames_g counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0
1	txgbfrmim	MMC Transmit Good Bad Frame Counter Interrupt Mask Setting this bit masks the interrupt when the txframecount_gb counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0
0	txgboctim	MMC Transmit Good Bad Octet Counter Interrupt Mask Setting this bit masks the interrupt when the txoctetcount_gb counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0

gmacgrp_txoctetcount_gb

Register 69 (Transmit Octet Count for Good and Bad Frames)

This register maintains the number of bytes transmitted in good and bad frames exclusive of preamble and retried bytes.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800114
i_emac_emac1	0xFF802000	0xFF802114
i_emac_emac2	0xFF804000	0xFF804114

Offset: 0x114

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_txoctetcount_gb Fields

Bit	Name	Description	Access	Reset
31:0	cnt	This field indicates the number of bytes transmitted in good and bad frames exclusive of preamble and retried bytes.	RO	0x0

gmacgrp_txframecount_gb

Register 70 (Transmit Frame Count for Good and Bad Frames)

This register maintains the number of good and bad frames transmitted, exclusive of retried frames.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800118
i_emac_emac1	0xFF802000	0xFF802118
i_emac_emac2	0xFF804000	0xFF804118

Offset: 0x118

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_txframecount_gb Fields

Bit	Name	Description	Access	Reset
31:0	cnt	This field indicates the number of good and bad frames transmitted, exclusive of retried frames	RO	0x0

gmacgrp_txbroadcastframes_g

Register 71 (Transmit Frame Count for Good Broadcast Frames)

This register maintains the number of transmitted good broadcast frames.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF80011C
i_emac_emac1	0xFF802000	0xFF80211C
i_emac_emac2	0xFF804000	0xFF80411C

Offset: 0x11C

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_txbroadcastframes_g Fields

Bit	Name	Description	Access	Reset
31:0	cnt	This field indicates the number of transmitted good broadcast frames.	RO	0x0

gmacgrp_txmulticastframes_g

Register 72 (Transmit Frame Count for Good Multicast Frames)

This register maintains the number of transmitted good multicast frames.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800120
i_emac_emac1	0xFF802000	0xFF802120
i_emac_emac2	0xFF804000	0xFF804120

Offset: 0x120

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_txmulticastframes_g Fields

Bit	Name	Description	Access	Reset
31:0	cnt	This field indicates the number of transmitted good multicast frames.	RO	0x0

gmacgrp_tx64octets_gb

Register 73 (Transmit Octet Count for Good and Bad 64 Byte Frames)

This register maintains the number of transmitted good and bad frames with length of 64 bytes, exclusive of preamble and retried frames.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800124
i_emac_emac1	0xFF802000	0xFF802124
i_emac_emac2	0xFF804000	0xFF804124

Offset: 0x124

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_tx64octets_gb Fields

Bit	Name	Description	Access	Reset
31:0	cnt	This field indicates the number of transmitted good and bad frames with length of 64 bytes, exclusive of preamble and retried frames.	RO	0x0

gmacgrp_tx65to127octets_gb

Register 74 (Transmit Octet Count for Good and Bad 65 to 127 Bytes Frames)

This register maintains the number of transmitted good and bad frames with length between 65 and 127 (inclusive) bytes, exclusive of preamble and retried frames.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800128
i_emac_emac1	0xFF802000	0xFF802128
i_emac_emac2	0xFF804000	0xFF804128

Offset: 0x128

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_tx65to127octets_gb Fields

Bit	Name	Description	Access	Reset
31:0	cnt	This field indicates the number of transmitted good and bad frames with length between 65 and 127 (inclusive) bytes, exclusive of preamble and retried frames.	RO	0x0

gmacgrp_tx128to255octets_gb

Register 75 (Transmit Octet Count for Good and Bad 128 to 255 Bytes Frames)

This register maintains the number of transmitted good and bad frames with length between 128 and 255 (inclusive) bytes, exclusive of preamble and retried frames.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF80012C
i_emac_emac1	0xFF802000	0xFF80212C
i_emac_emac2	0xFF804000	0xFF80412C

Offset: 0x12C

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_tx128to255octets_gb Fields

Bit	Name	Description	Access	Reset
31:0	cnt	This field indicates the number of transmitted good and bad frames with length between 128 and 255 (inclusive) bytes, exclusive of preamble and retried frames.	RO	0x0

gmacgrp_tx256to511octets_gb

Register 76 (Transmit Octet Count for Good and Bad 256 to 511 Bytes Frames)

This register maintains the number of transmitted good and bad frames with length between 256 and 511 (inclusive) bytes, exclusive of preamble and retried frames.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800130
i_emac_emac1	0xFF802000	0xFF802130
i_emac_emac2	0xFF804000	0xFF804130

Offset: 0x130

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_tx256to511octets_gb Fields

Bit	Name	Description	Access	Reset
31:0	cnt	This field indicates the number of transmitted good and bad frames with length between 256 and 511 (inclusive) bytes, exclusive of preamble and retried frames.	RO	0x0

gmacgrp_tx512to1023octets_gb

Register 77 (Transmit Octet Count for Good and Bad 512 to 1023 Bytes Frames)

This register maintains the number of transmitted good and bad frames with length between 512 and 1,023 (inclusive) bytes, exclusive of preamble and retried frames.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800134
i_emac_emac1	0xFF802000	0xFF802134
i_emac_emac2	0xFF804000	0xFF804134

Offset: 0x134

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_tx512to1023octets_gb Fields

Bit	Name	Description	Access	Reset
31:0	cnt	This field indicates the number of transmitted good and bad frames with length between 512 and 1,023 (inclusive) bytes, exclusive of preamble and retried frames.	RO	0x0

gmacgrp_tx1024tomaxoctets_gb

Register 78 (Transmit Octet Count for Good and Bad 1024 to Maxsize Bytes Frames)

This register maintains the number of transmitted good and bad frames with length between 1,024 and maxsize (inclusive) bytes, exclusive of preamble and retried frames.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800138
i_emac_emac1	0xFF802000	0xFF802138
i_emac_emac2	0xFF804000	0xFF804138

Offset: 0x138

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_tx1024tomaxoctets_gb Fields

Bit	Name	Description	Access	Reset
31:0	cnt	This field indicates the number of good and bad frames transmitted with length between 1,024 and maxsize (inclusive) bytes, exclusive of preamble and retried frames.	RO	0x0

gmacgrp_txunicastframes_gb

Register 79 (Transmit Frame Count for Good and Bad Unicast Frames)

This register maintains the number of transmitted good and bad unicast frames.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF80013C
i_emac_emac1	0xFF802000	0xFF80213C
i_emac_emac2	0xFF804000	0xFF80413C

Offset: 0x13C

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_txunicastframes_gb Fields

Bit	Name	Description	Access	Reset
31:0	cnt	This field indicates the number of transmitted good and bad unicast frames.	RO	0x0

gmacgrp_txmulticastframes_gb

Register 80 (Transmit Frame Count for Good and Bad Multicast Frames)

This register maintains the number of transmitted good and bad multicast frames.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800140
i_emac_emac1	0xFF802000	0xFF802140
i_emac_emac2	0xFF804000	0xFF804140

Offset: 0x140

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_txmulticastframes_gb Fields

Bit	Name	Description	Access	Reset
31:0	cnt	This field indicates the number of transmitted good and bad multicast frames.	RO	0x0

gmacgrp_txbroadcastframes_gb

Register 81 (Transmit Frame Count for Good and Bad Broadcast Frames)

This register maintains the number of transmitted good and bad broadcast frames.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800144
i_emac_emac1	0xFF802000	0xFF802144
i_emac_emac2	0xFF804000	0xFF804144

Offset: 0x144

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_txbroadcastframes_gb Fields

Bit	Name	Description	Access	Reset
31:0	cnt	This field indicates the number of transmitted good and bad broadcast frames.	RO	0x0

gmacgrp_txunderflowerror

Register 82 (Transmit Frame Count for Underflow Error Frames)

This register maintains the number of frames aborted because of frame underflow error.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800148
i_emac_emac1	0xFF802000	0xFF802148
i_emac_emac2	0xFF804000	0xFF804148

Offset: 0x148

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_txunderflowerror Fields

Bit	Name	Description	Access	Reset
31:0	cnt	This field indicates the number of frames aborted because of frame underflow error.	RO	0x0

gmacgrp_txsinglecol_g

Register 83 (Transmit Frame Count for Frames Transmitted after Single Collision)

This register maintains the number of successfully transmitted frames after a single collision in the half-duplex mode.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF80014C
i_emac_emac1	0xFF802000	0xFF80214C
i_emac_emac2	0xFF804000	0xFF80414C

Offset: 0x14C

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_txsinglecol_g Fields

Bit	Name	Description	Access	Reset
31:0	cnt	This field indicates the number of successfully transmitted frames after a single collision in the half-duplex mode.	RO	0x0

gmacgrp_txmulticol_g

Register 84 (Transmit Frame Count for Frames Transmitted after Multiple Collision)

This register maintains the number of successfully transmitted frames after multiple collisions in the half-duplex mode.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800150
i_emac_emac1	0xFF802000	0xFF802150
i_emac_emac2	0xFF804000	0xFF804150

Offset: 0x150

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_txmulticol_g Fields

Bit	Name	Description	Access	Reset
31:0	cnt	This field indicates the number of successfully transmitted frames after multiple collisions in the half-duplex mode.	RO	0x0

gmacgrp_txdeferred

Register 85 (Transmit Frame Count for Deferred Frames)

This register maintains the number of successfully transmitted frames after a deferral in the half-duplex mode.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800154
i_emac_emac1	0xFF802000	0xFF802154
i_emac_emac2	0xFF804000	0xFF804154

Offset: 0x154

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_txdeferred Fields

Bit	Name	Description	Access	Reset
31:0	cnt	This field indicates the number of successfully transmitted frames after a deferral in the half-duplex mode.	RO	0x0

gmacgrp_txlatecol

Register 86 (Transmit Frame Count for Late Collision Error Frames)

This register maintains the number of frames aborted because of late collision error.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800158
i_emac_emac1	0xFF802000	0xFF802158
i_emac_emac2	0xFF804000	0xFF804158

Offset: 0x158

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_txlatecol Fields

Bit	Name	Description	Access	Reset
31:0	cnt	This field indicates the number of frames aborted because of late collision error.	RO	0x0

gmacgrp_txexesscol

Register 87 (Transmit Frame Count for Excessive Collision Error Frames)

This register maintains the number of frames aborted because of excessive (16) collision error.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF80015C
i_emac_emac1	0xFF802000	0xFF80215C
i_emac_emac2	0xFF804000	0xFF80415C

Offset: 0x15C

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_txexesscol Fields

Bit	Name	Description	Access	Reset
31:0	cnt	This field indicates the number of frames aborted because of excessive (16) collision error.	RO	0x0

gmacgrp_txcarriererr

Register 88 (Transmit Frame Count for Carrier Sense Error Frames)

This register maintains the number of frames aborted because of carrier sense error (no carrier or loss of carrier).

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800160
i_emac_emac1	0xFF802000	0xFF802160
i_emac_emac2	0xFF804000	0xFF804160

Offset: 0x160

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_txcarriererr Fields

Bit	Name	Description	Access	Reset
31:0	cnt	This field indicates the number of frames aborted because of carrier sense error (no carrier or loss of carrier).	RO	0x0

gmacgrp_txoctetcnt

Register 89 (Transmit Octet Count for Good Frames)

This register maintains the number of bytes transmitted, exclusive of preamble, in good frames.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800164
i_emac_emac1	0xFF802000	0xFF802164
i_emac_emac2	0xFF804000	0xFF804164

Offset: 0x164

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
txoctetcount_g RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
txoctetcount_g RO 0x0															

gmacgrp_txoctetcnt Fields

Bit	Name	Description	Access	Reset
31:0	txoctetcount_g	This field indicates the number of bytes transmitted, exclusive of preamble, in good frames.	RO	0x0

gmacgrp_txframecount_g

Register 90 (Transmit Frame Count for Good Frames)

This register maintains the number of transmitted good frames, exclusive of preamble.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800168
i_emac_emac1	0xFF802000	0xFF802168
i_emac_emac2	0xFF804000	0xFF804168

Offset: 0x168

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_txframecount_g Fields

Bit	Name	Description	Access	Reset
31:0	cnt	This field indicates the number of transmitted good frames, exclusive of preamble.	RO	0x0

gmacgrp_txexcessdef

Register 91 (Transmit Frame Count for Excessive Deferral Error Frames)

This register maintains the number of frames aborted because of excessive deferral error, that is, frames deferred for more than two max-sized frame times.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF80016C
i_emac_emac1	0xFF802000	0xFF80216C
i_emac_emac2	0xFF804000	0xFF80416C

Offset: 0x16C

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_txexcessdef Fields

Bit	Name	Description	Access	Reset
31:0	cnt	This field indicates the number of frames aborted because of excessive deferral error, that is, frames deferred for more than two max-sized frame times.	RO	0x0

gmacgrp_txpauseframes

Register 92 (Transmit Frame Count for Good PAUSE Frames)

This register maintains the number of transmitted good PAUSE frames.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800170
i_emac_emac1	0xFF802000	0xFF802170
i_emac_emac2	0xFF804000	0xFF804170

Offset: 0x170

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_txpauseframes Fields

Bit	Name	Description	Access	Reset
31:0	cnt	This field indicates the number of transmitted good PAUSE frames.	RO	0x0

gmacgrp_txvlanframes_g

Register 93 (Transmit Frame Count for Good VLAN Frames)

This register maintains the number of transmitted good VLAN frames, exclusive of retried frames.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800174
i_emac_emac1	0xFF802000	0xFF802174

Module Instance	Base Address	Register Address
i_emac_emac2	0xFF804000	0xFF804174

Offset: 0x174

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_txvlanframes_g Fields

Bit	Name	Description	Access	Reset
31:0	cnt	This register maintains the number of transmitted good VLAN frames, exclusive of retried frames.	RO	0x0

gmacgrp_txoversize_g

Register 94 (Transmit Frame Count for Good Oversize Frames)

This register maintains the number of transmitted good Oversize frames, exclusive of retried frames.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800178
i_emac_emac1	0xFF802000	0xFF802178
i_emac_emac2	0xFF804000	0xFF804178

Offset: 0x178

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_txoversize_g Fields

Bit	Name	Description	Access	Reset
31:0	cnt	This field indicates the number of frames transmitted without errors and with length greater than the maxsize (1,518 or 1,522 bytes for VLAN tagged frames; 2000 bytes if enabled in bit 27 of Register 0 (MAC Configuration Register)).	RO	0x0

gmacgrp_rxframecount_gb

Register 96 (Receive Frame Count for Good and Bad Frames)

This register maintains the number of received good and bad frames.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800180
i_emac_emac1	0xFF802000	0xFF802180
i_emac_emac2	0xFF804000	0xFF804180

Offset: 0x180

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_rxframecount_gb Fields

Bit	Name	Description	Access	Reset
31:0	cnt	This field indicates the number of received good and bad frames.	RO	0x0

gmacgrp_rxoctetcount_gb

Register 97 (Receive Octet Count for Good and Bad Frames)

This register maintains the number of bytes received, exclusive of preamble, in good and bad frames.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800184
i_emac_emac1	0xFF802000	0xFF802184
i_emac_emac2	0xFF804000	0xFF804184

Offset: 0x184

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_rxoctetcount_gb Fields

Bit	Name	Description	Access	Reset
31:0	cnt	This field indicates the number of bytes received, exclusive of preamble, in good and bad frames.	RO	0x0

gmacgrp_rxoctetcount_g

Register 98 (Receive Octet Count for Good Frames)

This register maintains the number of bytes received, exclusive of preamble, only in good frames.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800188
i_emac_emac1	0xFF802000	0xFF802188
i_emac_emac2	0xFF804000	0xFF804188

Offset: 0x188

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_rxoctetcount_g Fields

Bit	Name	Description	Access	Reset
31:0	cnt	This field indicates the number of bytes received, exclusive of preamble, only in good frames.	RO	0x0

gmacgrp_rxbroadcastframes_g

Register 99 (Receive Frame Count for Good Broadcast Frames)

This register maintains the number of received good broadcast frames.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF80018C
i_emac_emac1	0xFF802000	0xFF80218C
i_emac_emac2	0xFF804000	0xFF80418C

Offset: 0x18C

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_rxbroadcastframes_g Fields

Bit	Name	Description	Access	Reset
31:0	cnt	This field indicates the number of received good broadcast frames.	RO	0x0

gmacgrp_rxmulticastframes_g

Register 100 (Receive Frame Count for Good Multicast Frames)

This register maintains the number of received good multicast frames.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800190

Module Instance	Base Address	Register Address
i_emac_emac1	0xFF802000	0xFF802190
i_emac_emac2	0xFF804000	0xFF804190

Offset: 0x190

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_rxmulticastframes_g Fields

Bit	Name	Description	Access	Reset
31:0	cnt	This field indicates the number of received good multicast frames.	RO	0x0

gmacgrp_rxcrcerror

Register 101 (Receive Frame Count for CRC Error Frames)

This register maintains the number of frames received with CRC error.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800194
i_emac_emac1	0xFF802000	0xFF802194
i_emac_emac2	0xFF804000	0xFF804194

Offset: 0x194

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_rxcrcerror Fields

Bit	Name	Description	Access	Reset
31:0	cnt	This field indicates the number of frames received with CRC error.	RO	0x0

gmacgrp_rxalignmenterror

Register 102 (Receive Frame Count for Alignment Error Frames)

This register maintains the number of frames received with alignment (dribble) error. This field is valid only in the 10 or 100 Mbps mode.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800198
i_emac_emac1	0xFF802000	0xFF802198
i_emac_emac2	0xFF804000	0xFF804198

Offset: 0x198

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_rxalignmenterror Fields

Bit	Name	Description	Access	Reset
31:0	cnt	This field indicates the number of frames received with alignment (dribble) error. This field is valid only in the 10 or 100 Mbps mode.	RO	0x0

gmacgrp_rxrunterror

Register 103 (Receive Frame Count for Runt Error Frames)

This register maintains the number of frames received with runt error(<64 bytes and CRC error).

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF80019C
i_emac_emac1	0xFF802000	0xFF80219C
i_emac_emac2	0xFF804000	0xFF80419C

Offset: 0x19C

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_rxrunterror Fields

Bit	Name	Description	Access	Reset
31:0	cnt	This field indicates the number of frames received with runt error(<64 bytes and CRC error).	RO	0x0

gmacgrp_rxjabbererror

Register 104 (Receive Frame Count for Jabber Error Frames)

This register maintains the number of giant frames received with length (including CRC) greater than 1,518 bytes (1,522 bytes for VLAN tagged) and with CRC error. If Jumbo Frame mode is enabled, then frames of length greater than 9,018 bytes (9,022 for VLAN tagged) are considered as giant frames.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8001A0
i_emac_emac1	0xFF802000	0xFF8021A0
i_emac_emac2	0xFF804000	0xFF8041A0

Offset: 0x1A0

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_rxjabbererror Fields

Bit	Name	Description	Access	Reset
31:0	cnt	This field indicates the number of giant frames received with length (including CRC) greater than 1,518 bytes (1,522 bytes for VLAN tagged) and with CRC error. If Jumbo Frame mode is enabled, then frames of length greater than 9,018 bytes (9,022 for VLAN tagged) are considered as giant frames.	RO	0x0

gmacgrp_rxundersize_g

Register 105 (Receive Frame Count for Undersize Frames)

This register maintains the number of frames received with length less than 64 bytes and without errors.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8001A4
i_emac_emac1	0xFF802000	0xFF8021A4
i_emac_emac2	0xFF804000	0xFF8041A4

Offset: 0x1A4

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_rxundersize_g Fields

Bit	Name	Description	Access	Reset
31:0	cnt	This field indicates the number of frames received with length less than 64 bytes and without errors.	RO	0x0

gmacgrp_rxoversize_g

Register 106 (Receive Frame Count for Oversize Frames)

This register maintains the number of frames received with length greater than the maxsize (1,518 or 1,522 for VLAN tagged frames) and without errors.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8001A8
i_emac_emac1	0xFF802000	0xFF8021A8
i_emac_emac2	0xFF804000	0xFF8041A8

Offset: 0x1A8

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_rxoversize_g Fields

Bit	Name	Description	Access	Reset
31:0	cnt	This field indicates the number of frames received without errors, with length greater than the maxsize (1,518 or 1,522 for VLAN tagged frames; 2,000 bytes if enabled in bit 27 of Register 0 (MAC Configuration Register)).	RO	0x0

gmacgrp_rx64octets_gb

Register 107 (Receive Frame Count for Good and Bad 64 Byte Frames)

This register maintains the number of received good and bad frames with length 64 bytes, exclusive of preamble.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8001AC
i_emac_emac1	0xFF802000	0xFF8021AC
i_emac_emac2	0xFF804000	0xFF8041AC

Offset: 0x1AC

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_rx64octets_gb Fields

Bit	Name	Description	Access	Reset
31:0	cnt	This field indicates the number of received good and bad frames with length 64 bytes, exclusive of preamble.	RO	0x0

gmacgrp_rx65to127octets_gb

Register 108 (Receive Frame Count for Good and Bad 65 to 127 Bytes Frames)

This register maintains the number of received good and bad frames received with length between 65 and 127 (inclusive) bytes, exclusive of preamble.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8001B0
i_emac_emac1	0xFF802000	0xFF8021B0
i_emac_emac2	0xFF804000	0xFF8041B0

Offset: 0x1B0

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_rx65to127octets_gb Fields

Bit	Name	Description	Access	Reset
31:0	cnt	This field indicates the number of received good and bad frames received with length between 65 and 127 (inclusive) bytes, exclusive of preamble.	RO	0x0

gmacgrp_rx128to255octets_gb

Register 109 (Receive Frame Count for Good and Bad 128 to 255 Bytes Frames)

This register maintains the number of received good and bad frames with length between 128 and 255 (inclusive) bytes, exclusive of preamble.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8001B4
i_emac_emac1	0xFF802000	0xFF8021B4
i_emac_emac2	0xFF804000	0xFF8041B4

Offset: 0x1B4

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_rx128to255octets_gb Fields

Bit	Name	Description	Access	Reset
31:0	cnt	This field indicates the number of received good and bad frames with length between 128 and 255 (inclusive) bytes, exclusive of preamble.	RO	0x0

gmacgrp_rx256to511octets_gb

Register 110 (Receive Frame Count for Good and Bad 256 to 511 Bytes Frames)

This register maintains the number of received good and bad frames with length between 256 and 511 (inclusive) bytes, exclusive of preamble.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8001B8
i_emac_emac1	0xFF802000	0xFF8021B8
i_emac_emac2	0xFF804000	0xFF8041B8

Offset: 0x1B8

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_rx256to511octets_gb Fields

Bit	Name	Description	Access	Reset
31:0	cnt	This field indicates the number of received good and bad frames with length between 256 and 511 (inclusive) bytes, exclusive of preamble.	RO	0x0

gmacgrp_rx512to1023octets_gb

Register 111 (Receive Frame Count for Good and Bad 512 to 1,023 Bytes Frames)

This register maintains the number of received good and bad frames with length between 512 and 1,023 (inclusive) bytes, exclusive of preamble.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8001BC
i_emac_emac1	0xFF802000	0xFF8021BC
i_emac_emac2	0xFF804000	0xFF8041BC

Offset: 0x1BC

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_rx512to1023octets_gb Fields

Bit	Name	Description	Access	Reset
31:0	cnt	This field indicates the number of received good and bad frames with length between 512 and 1,023 (inclusive) bytes, exclusive of preamble.	RO	0x0

gmacgrp_rx1024tomaxoctets_gb

Register 112 (Receive Frame Count for Good and Bad 1,024 to Maxsize Bytes Frames)

This register maintains the number of received good and bad frames with length between 1,024 and maxsize (inclusive) bytes, exclusive of preamble.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8001C0
i_emac_emac1	0xFF802000	0xFF8021C0
i_emac_emac2	0xFF804000	0xFF8041C0

Offset: 0x1C0

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_rx1024tomaxoctets_gb Fields

Bit	Name	Description	Access	Reset
31:0	cnt	This field indicates the number of received good and bad frames with length between 1,024 and maxsize (inclusive) bytes, exclusive of preamble and retried frames.	RO	0x0

gmacgrp_rxunicastframes_g

Register 113 (Receive Frame Count for Good Unicast Frames)

This register maintains the number of received good unicast frames.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8001C4
i_emac_emac1	0xFF802000	0xFF8021C4
i_emac_emac2	0xFF804000	0xFF8041C4

Offset: 0x1C4

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_rxunicastframes_g Fields

Bit	Name	Description	Access	Reset
31:0	cnt	This field indicates the number of received good unicast frames.	RO	0x0

gmacgrp_rxlengtherror

Register 114 (Receive Frame Count for Length Error Frames)

This register maintains the number of frames received with length error (Length type field not equal to frame size) for all frames with valid length field.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8001C8
i_emac_emac1	0xFF802000	0xFF8021C8
i_emac_emac2	0xFF804000	0xFF8041C8

Offset: 0x1C8

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_rxlengtherror Fields

Bit	Name	Description	Access	Reset
31:0	cnt	This field indicates the number of frames received with length error (Length type field not equal to frame size) for all frames with valid length field.	RO	0x0

gmacgrp_rxoutofrangetype

Register 115 (Receive Frame Count for Out of Range Frames)

This register maintains the number of received frames with length field not equal to the valid frame size (greater than 1,500 but less than 1,536).

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8001CC
i_emac_emac1	0xFF802000	0xFF8021CC
i_emac_emac2	0xFF804000	0xFF8041CC

Offset: 0x1CC

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_rxoutofrangetype Fields

Bit	Name	Description	Access	Reset
31:0	cnt	This field indicates the number of received frames with length field not equal to the valid frame size (greater than 1,500 but less than 1,536).	RO	0x0

gmacgrp_rxpauseframes

Register 116 (Receive Frame Count for PAUSE Frames)

This register maintains the number of received good and valid PAUSE frames.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8001D0
i_emac_emac1	0xFF802000	0xFF8021D0
i_emac_emac2	0xFF804000	0xFF8041D0

Offset: 0x1D0

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_rxpauseframes Fields

Bit	Name	Description	Access	Reset
31:0	cnt	This field indicates the number of received good and valid PAUSE frames.	RO	0x0

gmacgrp_rxfifooverflow

Register 117 (Receive Frame Count for FIFO Overflow Frames)

This register maintains the number of received frames missed because of FIFO overflow.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8001D4
i_emac_emac1	0xFF802000	0xFF8021D4
i_emac_emac2	0xFF804000	0xFF8041D4

Offset: 0x1D4

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_rxfifooverflow Fields

Bit	Name	Description	Access	Reset
31:0	cnt	This field indicates the number of received frames missed because of FIFO overflow.	RO	0x0

gmacgrp_rxvlanframes_gb

Register 118 (Receive Frame Count for Good and Bad VLAN Frames)

This register maintains the number of received good and bad VLAN frames.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8001D8
i_emac_emac1	0xFF802000	0xFF8021D8
i_emac_emac2	0xFF804000	0xFF8041D8

Offset: 0x1D8

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_rxvlanframes_gb Fields

Bit	Name	Description	Access	Reset
31:0	cnt	This field indicates the number of received good and bad VLAN frames.	RO	0x0

gmacgrp_rxwatchdogerror

Register 119 (Receive Frame Count for Watchdog Error Frames)

This register maintains the number of frames received with error because of the watchdog timeout error (frames with more than 2,048 bytes or value programmed in Register 55 (Watchdog Timeout Register)).

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8001DC
i_emac_emac1	0xFF802000	0xFF8021DC
i_emac_emac2	0xFF804000	0xFF8041DC

Offset: 0x1DC

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_rxwatchdogerror Fields

Bit	Name	Description	Access	Reset
31:0	cnt	This field indicates the number of frames received with error because of the watchdog timeout error (frames with more than 2,048 bytes or value programmed in Register 55 (Watchdog Timeout Register)).	RO	0x0

gmacgrp_rxrcverror

Register 120 (Receive Frame Count for Receive Error Frames)
This register maintains the number of frames received with error because of the GMII/MII RXER error.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8001E0
i_emac_emac1	0xFF802000	0xFF8021E0
i_emac_emac2	0xFF804000	0xFF8041E0

Offset: 0x1E0

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_rxrcverror Fields

Bit	Name	Description	Access	Reset
31:0	cnt	This field indicates the number of frames received with error because of the GMII/MII RXER error or Frame Extension error on GMII.	RO	0x0

gmacgrp_rxctrlframes_g

Register 121 (Receive Frame Count for Good Control Frames Frames)

This register maintains the number of good control frames received.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8001E4

Module Instance	Base Address	Register Address
i_emac_emac1	0xFF802000	0xFF8021E4
i_emac_emac2	0xFF804000	0xFF8041E4

Offset: 0x1E4

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_rxctrlframes_g Fields

Bit	Name	Description	Access	Reset
31:0	cnt	This field indicates the number of good control frames received.	RO	0x0

gmacgrp_mmc_ipc_receive_interrupt_mask

This register maintains the mask for the interrupt generated from the receive IPC statistic counters.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800200
i_emac_emac1	0xFF802000	0xFF802200
i_emac_emac2	0xFF804000	0xFF804200

Offset: 0x200

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		rxicmperoim RW 0x0	rxicmpgoim RW 0x0	rxtcperoim RW 0x0	rxtcpgoim RW 0x0	rxudperoim RW 0x0	rxudpgoim RW 0x0	rxipv6nopayoim RW 0x0	rxipv6heroim RW 0x0	rxipv6goim RW 0x0	rxipv4udsbloim RW 0x0	rxipv4fragoim RW 0x0	rxipv4nopayoim RW 0x0	rxipv4heroim RW 0x0	rxipv4goim RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		rxicmperfim RW 0x0	rxicmpgfim RW 0x0	rxtcperfim RW 0x0	rxtcpgfim RW 0x0	rxudperfim RW 0x0	rxudpgfim RW 0x0	rxipv6nopayfim RW 0x0	rxipv6herfim RW 0x0	rxipv6gfim RW 0x0	rxipv4udsblfim RW 0x0	rxipv4fragfim RW 0x0	rxipv4nopayfim RW 0x0	rxipv4herfim RW 0x0	rxipv4gfim RW 0x0

gmacgrp_mmc_ipc_receive_interrupt_mask Fields

Bit	Name	Description	Access	Reset
29	rxicmperoim	Setting this bit masks the interrupt when the rxicmp_err_octets counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0
28	rxicmpgoim	Setting this bit masks the interrupt when the rxicmp_gd_octets counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0
27	rxtcperoim	Setting this bit masks the interrupt when the rxtcp_err_octets counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0

Bit	Name	Description	Access	Reset
26	rxtcpgoim	Setting this bit masks the interrupt when the rxtcp_gd_octets counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0
25	rxudperoim	Setting this bit masks the interrupt when the rxudp_err_octets counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0
24	rxudpgoim	Setting this bit masks the interrupt when the rxudp_gd_octets counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0
23	rxipv6nopayoim	Setting this bit masks the interrupt when the rxipv6_nopay_ octets counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0
22	rxipv6heroim	Setting this bit masks interrupt when the rxipv6_hdrerr_octets counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0

Bit	Name	Description	Access	Reset
21	rxipv6goim	Setting this bit masks the interrupt when the rxipv6_gd_octets counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0
20	rxipv4udsbloim	Setting this bit masks the interrupt when the rxipv4_udsbl_ octets counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0
19	rxipv4fragoim	Setting this bit masks the interrupt when the rxipv4_frag_ octets counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0
18	rxipv4nopayoim	Setting this bit masks the interrupt when the rxipv4_nopay_ octets counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0
17	rxipv4heroim	Setting this bit masks the interrupt when the rxipv4_hdrerr_ octets counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0

Bit	Name	Description	Access	Reset
16	rxipv4goim	Setting this bit masks the interrupt when the rxipv4_gd_octets counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0
13	rxicmperfim	Setting this bit masks the interrupt when the rxicmp_err_frms counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0
12	rxicmpgfim	Setting this bit masks the interrupt when the rxicmp_gd_frms counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0
11	rxtcperfim	Setting this bit masks the interrupt when the rxtcp_err_frms counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0
10	rxtcpgfim	Setting this bit masks the interrupt when the rxtcp_gd_frms counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0

Bit	Name	Description	Access	Reset
9	rxudperfim	Setting this bit masks the interrupt when the rxudp_err_frms counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0
8	rxudpgfim	Setting this bit masks the interrupt when the rxudp_gd_frms counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0
7	rxipv6nopayfim	Setting this bit masks the interrupt when the rxipv6_nopay_ frms counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0
6	rxipv6herfim	Setting this bit masks the interrupt when the rxipv6_hdrerr_ frms counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0
5	rxipv6gfim	Setting this bit masks the interrupt when the rxipv6_gd_frms counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0

Bit	Name	Description	Access	Reset
4	rxipv4udsblfim	Setting this bit masks the interrupt when the rxipv4_udsbl_frms counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0
3	rxipv4fragfim	Setting this bit masks the interrupt when the rxipv4_frag_frms counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0
2	rxipv4nopayfim	Setting this bit masks the interrupt when the rxipv4_nopay_frms counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0
1	rxipv4herfim	Setting this bit masks the interrupt when the rxipv4_hdrerr_frms counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOMASKINTR 0x1 MASKINTR	RW	0x0

Bit	Name	Description	Access	Reset
0	rxipv4gfim	Setting this bit masks the interrupt when the rxipv4_gd_frms counter reaches half of the maximum value or the maximum value. Value: Description 0x0: NOMASKINTR 0x1: MASKINTR	RW	0x0

gmacgrp_mmc_ipc_receive_interrupt

This register maintains the interrupts generated when receive IPC statistic counters reach half their maximum values (0x8000_0000 for 32-bit counter and 0x8000 for 16-bit counter), and when they cross their maximum values (0xFFFF_FFFF for 32-bit counter and 0xFFFF for 16-bit counter). When Counter Stop Rollover is set, then interrupts are set but the counter remains at all-ones. The MMC Receive Checksum Offload Interrupt register is 32-bits wide. When the MMC IPC counter that caused the interrupt is read, its corresponding interrupt bit is cleared. The counter's least-significant byte lane (bits[7:0]) must be read to clear the interrupt bit.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800208
i_emac_emac1	0xFF802000	0xFF802208
i_emac_emac2	0xFF804000	0xFF804208

Offset: 0x208

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		rxicmperois RO 0x0	rxicmpgois RO 0x0	rxtcperois RO 0x0	rxtcpgois RO 0x0	rxudperois RO 0x0	rxudpgois RO 0x0	rxipv6nopayois RO 0x0	rxipv6heroris RO 0x0	rxipv6gois RO 0x0	rxipv4udsblois RO 0x0	rxipv4fragois RO 0x0	rxipv4nopayois RO 0x0	rxipv4heroris RO 0x0	rxipv4gois RO 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		rxicmperfis RO 0x0	rxicmpgfis RO 0x0	rxtcperfis RO 0x0	rxtcpgfis RO 0x0	rxudperfis RO 0x0	rxudpgfis RO 0x0	rxipv6nopayfis RO 0x0	rxipv6herfis RO 0x0	rxipv6gfis RO 0x0	rxipv4udsblfis RO 0x0	rxipv4fragfis RO 0x0	rxipv4nopayfis RO 0x0	rxipv4herfis RO 0x0	rxipv4gfis RO 0x0

gmacgrp_mmc_ipc_receive_interrupt Fields

Bit	Name	Description	Access	Reset
29	rxicmperois	This bit is set when the rxicmp_err_octets counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOINTERRUPT 0x1 INTERR	RO	0x0
28	rxicmpgois	This bit is set when the rxicmp_gd_octets counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOINTERRUPT 0x1 INTERR	RO	0x0
27	rxtcperois	This bit is set when the rxtcp_err_octets counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOINTERRUPT 0x1 INTERR	RO	0x0

Bit	Name	Description	Access	Reset
26	rxtcpgois	This bit is set when the rxtcp_gd_octets counter reaches half the maximum value or the maximum value. **Value** **Description** 0x0 NOINTERRUPT 0x1 INTERR	RO	0x0
25	rxudperois	This bit is set when the rxudp_err_octets counter reaches half the maximum value or the maximum value. **Value** **Description** 0x0 NOINTERRUPT 0x1 INTERR	RO	0x0
24	rxudpgois	This bit is set when the rxudp_gd_octets counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOINTERRUPT 0x1 INTERR	RO	0x0
23	rxipv6nopayois	This bit is set when the rxipv6_nopay_octets counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOINTERRUPT 0x1 INTERR	RO	0x0
22	rxipv6herois	This bit is set when the rxipv6_hdrerr_octets counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOINTERRUPT 0x1 INTERR	RO	0x0

Bit	Name	Description	Access	Reset
21	rxipv6gois	This bit is set when the rxipv6_gd_octets counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOINTERRUPT 0x1 INTERR	RO	0x0
20	rxipv4udsblois	This bit is set when the rxipv4_udsbl_octets counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOINTERRUPT 0x1 INTERR	RO	0x0
19	rxipv4fragois	This bit is set when the rxipv4_frag_octets counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOINTERRUPT 0x1 INTERR	RO	0x0
18	rxipv4nopayois	This bit is set when the rxipv4_nopay_octets counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOINTERRUPT 0x1 INTERR	RO	0x0
17	rxipv4herois	This bit is set when the rxipv4_hdrerr_octets counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOINTERRUPT 0x1 INTERR	RO	0x0

Bit	Name	Description	Access	Reset
16	rxipv4gois	This bit is set when the rxipv4_gd_octets counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOINTERRUPT 0x1 INTERR	RO	0x0
13	rxicmperfis	This bit is set when the rxicmp_err_frms counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOINTERRUPT 0x1 INTERR	RO	0x0
12	rxicmpgfis	This bit is set when the rxicmp_gd_frms counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOINTERRUPT 0x1 INTERR	RO	0x0
11	rxtcperfis	This bit is set when the rxtcp_err_frms counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOINTERRUPT 0x1 INTERR	RO	0x0
10	rxtcpgfis	This bit is set when the rxtcp_gd_frms counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOINTERRUPT 0x1 INTERR	RO	0x0

Bit	Name	Description	Access	Reset
9	rxudperfis	This bit is set when the rxudp_err_frms counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOINTERRUPT 0x1 INTERR	RO	0x0
8	rxudpgfis	This bit is set when the rxudp_gd_frms counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOINTERRUPT 0x1 INTERR	RO	0x0
7	rxipv6nopayfis	This bit is set when the rxipv6_nopay_frms counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOINTERRUPT 0x1 INTERR	RO	0x0
6	rxipv6herfis	This bit is set when the rxipv6_hdrerr_frms counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOINTERRUPT 0x1 INTERR	RO	0x0
5	rxipv6gfis	This bit is set when the rxipv6_gd_frms counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOINTERRUPT 0x1 INTERR	RO	0x0

Bit	Name	Description	Access	Reset
4	rxipv4udsblfis	This bit is set when the rxipv4_udsbl_frms counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOINTERRUPT 0x1 INTERR	RO	0x0
3	rxipv4fragfis	This bit is set when the rxipv4_frag_frms counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOINTERRUPT 0x1 INTERR	RO	0x0
2	rxipv4nopayfis	This bit is set when the rxipv4_nopay_frms counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOINTERRUPT 0x1 INTERR	RO	0x0
1	rxipv4herfis	This bit is set when the rxipv4_hdrerr_frms counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOINTERRUPT 0x1 INTERR	RO	0x0
0	rxipv4gfis	This bit is set when the rxipv4_gd_frms counter reaches half of the maximum value or the maximum value. **Value** **Description** 0x0 NOINTERRUPT 0x1 INTERR	RO	0x0

gmacgrp_rxipv4_gd_frms

Number of good IPv4 datagrams received with the TCP, UDP, or ICMP payload

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800210
i_emac_emac1	0xFF802000	0xFF802210
i_emac_emac2	0xFF804000	0xFF804210

Offset: 0x210

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_rxipv4_gd_frms Fields

Bit	Name	Description	Access	Reset
31:0	cnt	Number of good IPv4 datagrams received with the TCP, UDP, or ICMP payload	RO	0x0

gmacgrp_rxipv4_hdrerr_frms

Number of IPv4 datagrams received with header (checksum, length, or version mismatch) errors

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800214
i_emac_emac1	0xFF802000	0xFF802214
i_emac_emac2	0xFF804000	0xFF804214

Offset: 0x214

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_rxipv4_hdrerr_frms Fields

Bit	Name	Description	Access	Reset
31:0	cnt	Number of IPv4 datagrams received with header (checksum, length, or version mismatch) errors	RO	0x0

gmacgrp_rxipv4_nopay_frms

Number of IPv4 datagram frames received that did not have a TCP, UDP, or ICMP payload processed by the Checksum engine

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800218
i_emac_emac1	0xFF802000	0xFF802218
i_emac_emac2	0xFF804000	0xFF804218

Offset: 0x218

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_rxipv4_nopay_frms Fields

Bit	Name	Description	Access	Reset
31:0	cnt	Number of IPv4 datagram frames received that did not have a TCP, UDP, or ICMP payload processed by the Checksum engine	RO	0x0

gmacgrp_rxipv4_frag_frms

Number of good IPv4 datagrams with fragmentation

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF80021C
i_emac_emac1	0xFF802000	0xFF80221C
i_emac_emac2	0xFF804000	0xFF80421C

Offset: 0x21C

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_rxipv4_frag_frms Fields

Bit	Name	Description	Access	Reset
31:0	cnt	Number of good IPv4 datagrams with fragmentation	RO	0x0

gmacgrp_rxipv4_udsbl_frms

Number of good IPv4 datagrams received that had a UDP

payload with checksum disabled

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800220
i_emac_emac1	0xFF802000	0xFF802220
i_emac_emac2	0xFF804000	0xFF804220

Offset: 0x220

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_rxipv4_udsbl_frms Fields

Bit	Name	Description	Access	Reset
31:0	cnt	Number of good IPv4 datagrams received that had a UDP payload with checksum disabled	RO	0x0

gmacgrp_rxipv6_gd_frms

Number of good IPv6 datagrams received with TCP, UDP, or ICMP payloads

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800224
i_emac_emac1	0xFF802000	0xFF802224
i_emac_emac2	0xFF804000	0xFF804224

Offset: 0x224

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_rxipv6_gd_frms Fields

Bit	Name	Description	Access	Reset
31:0	cnt	Number of good IPv6 datagrams received with TCP, UDP, or ICMP payloads	RO	0x0

gmacgrp_rxipv6_hdrerr_frms

Number of IPv6 datagrams received with header errors (length or version mismatch)

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800228
i_emac_emac1	0xFF802000	0xFF802228
i_emac_emac2	0xFF804000	0xFF804228

Offset: 0x228

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_rxipv6_hdrerr_frms Fields

Bit	Name	Description	Access	Reset
31:0	cnt	Number of IPv6 datagrams received with header errors (length or version mismatch)	RO	0x0

gmacgrp_rxipv6_nopay_frms

Number of IPv6 datagram frames received that did not have a TCP, UDP, or ICMP payload. This includes all IPv6 datagrams with fragmentation or security extension headers

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF80022C
i_emac_emac1	0xFF802000	0xFF80222C
i_emac_emac2	0xFF804000	0xFF80422C

Offset: 0x22C

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_rxipv6_nopay_frms Fields

Bit	Name	Description	Access	Reset
31:0	cnt	Number of IPv6 datagram frames received that did not have a TCP, UDP, or ICMP payload. This includes all IPv6 datagrams with fragmentation or security extension headers	RO	0x0

gmacgrp_rxudp_gd_frms

Number of good IP datagrams with a good UDP payload. This counter is not updated when the counter is incremented

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800230
i_emac_emac1	0xFF802000	0xFF802230
i_emac_emac2	0xFF804000	0xFF804230

Offset: 0x230

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_rxudp_gd_frms Fields

Bit	Name	Description	Access	Reset
31:0	cnt	Number of good IP datagrams with a good UDP payload. This counter is not updated when the counter is incremented	RO	0x0

gmacgrp_rxudp_err_frms

Number of good IP datagrams whose UDP payload has a checksum error

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800234
i_emac_emac1	0xFF802000	0xFF802234

Module Instance	Base Address	Register Address
i_emac_emac2	0xFF804000	0xFF804234

Offset: 0x234

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_rxudp_err_frms Fields

Bit	Name	Description	Access	Reset
31:0	cnt	Number of good IP datagrams whose UDP payload has a checksum error	RO	0x0

gmacgrp_rxtcp_gd_frms

Number of good IP datagrams with a good TCP payload

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800238
i_emac_emac1	0xFF802000	0xFF802238
i_emac_emac2	0xFF804000	0xFF804238

Offset: 0x238

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_rxtcp_gd_frms Fields

Bit	Name	Description	Access	Reset
31:0	cnt	Number of good IP datagrams with a good TCP payload	RO	0x0

gmacgrp_rxtcp_err_frms

Number of good IP datagrams whose TCP payload has a checksum error

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF80023C
i_emac_emac1	0xFF802000	0xFF80223C
i_emac_emac2	0xFF804000	0xFF80423C

Offset: 0x23C

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_rxtcp_err_frms Fields

Bit	Name	Description	Access	Reset
31:0	cnt	Number of good IP datagrams whose TCP payload has a checksum error	RO	0x0

gmacgrp_rxicmp_gd_frms

Number of good IP datagrams with a good ICMP payload

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800240
i_emac_emac1	0xFF802000	0xFF802240
i_emac_emac2	0xFF804000	0xFF804240

Offset: 0x240

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_rxicmp_gd_frms Fields

Bit	Name	Description	Access	Reset
31:0	cnt	Number of good IP datagrams with a good ICMP payload	RO	0x0

gmacgrp_rxicmp_err_frms

Number of good IP datagrams whose ICMP payload has a checksum error

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800244
i_emac_emac1	0xFF802000	0xFF802244
i_emac_emac2	0xFF804000	0xFF804244

Offset: 0x244

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_rxicmp_err_frms Fields

Bit	Name	Description	Access	Reset
31:0	cnt	Number of good IP datagrams whose ICMP payload has a checksum error	RO	0x0

gmacgrp_rxipv4_gd_octets

Number of bytes received in good IPv4 datagrams encapsulating TCP, UDP, or ICMP data

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800250
i_emac_emac1	0xFF802000	0xFF802250
i_emac_emac2	0xFF804000	0xFF804250

Offset: 0x250

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_rxipv4_gd_octets Fields

Bit	Name	Description	Access	Reset
31:0	cnt	Number of bytes received in good IPv4 datagrams encapsulating TCP, UDP, or ICMP data	RO	0x0

gmacgrp_rxipv4_hdrerr_octets

Number of bytes received in IPv4 datagrams with header errors (checksum, length, version mismatch). The value in the Length field of IPv4 header is used to update this counter

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800254
i_emac_emac1	0xFF802000	0xFF802254
i_emac_emac2	0xFF804000	0xFF804254

Offset: 0x254

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_rxipv4_hdrerr_octets Fields

Bit	Name	Description	Access	Reset
31:0	cnt	Number of bytes received in IPv4 datagrams with header errors (checksum, length, version mismatch). The value in the Length field of IPv4 header is used to update this counter	RO	0x0

gmacgrp_rxipv4_nopay_octets

Number of bytes received in IPv4 datagrams that did not have a TCP, UDP, or ICMP payload. The value in the IPv4 headers Length field is used to update this counter

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800258
i_emac_emac1	0xFF802000	0xFF802258
i_emac_emac2	0xFF804000	0xFF804258

Offset: 0x258

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_rxipv4_nopay_octets Fields

Bit	Name	Description	Access	Reset
31:0	cnt	Number of bytes received in IPv4 datagrams that did not have a TCP, UDP, or ICMP payload. The value in the IPv4 headers Length field is used to update this counter	RO	0x0

gmacgrp_rxipv4_frag_octets

Number of bytes received in fragmented IPv4 datagrams. The value in the IPv4 headers Length field is used to update this counter

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF80025C
i_emac_emac1	0xFF802000	0xFF80225C
i_emac_emac2	0xFF804000	0xFF80425C

Offset: 0x25C

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_rxipv4_frag_octets Fields

Bit	Name	Description	Access	Reset
31:0	cnt	Number of bytes received in fragmented IPv4 datagrams. The value in the IPv4 headers Length field is used to update this counter	RO	0x0

gmacgrp_rxipv4_udsbl_octets

Number of bytes received in a UDP segment that had the UDP checksum disabled. This counter does not count IP Header bytes

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800260
i_emac_emac1	0xFF802000	0xFF802260

Module Instance	Base Address	Register Address
i_emac_emac2	0xFF804000	0xFF804260

Offset: 0x260

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_rxipv4_udsbl_octets Fields

Bit	Name	Description	Access	Reset
31:0	cnt	Number of bytes received in a UDP segment that had the UDP checksum disabled. This counter does not count IP Header bytes	RO	0x0

gmacgrp_rxipv6_gd_octets

Number of bytes received in good IPv6 datagrams encapsulating TCP, UDP or ICMPv6 data

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800264
i_emac_emac1	0xFF802000	0xFF802264
i_emac_emac2	0xFF804000	0xFF804264

Offset: 0x264

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_rxipv6_gd_octets Fields

Bit	Name	Description	Access	Reset
31:0	cnt	Number of bytes received in good IPv6 datagrams encapsulating TCP, UDP or ICMPv6 data	RO	0x0

gmacgrp_rxipv6_hdrerr_octets

Number of bytes received in IPv6 datagrams with header errors (length, version mismatch). The value in the IPv6 headers Length field is used to update this counter

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800268
i_emac_emac1	0xFF802000	0xFF802268
i_emac_emac2	0xFF804000	0xFF804268

Offset: 0x268

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_rxipv6_hdrerr_octets Fields

Bit	Name	Description	Access	Reset
31:0	cnt	Number of bytes received in IPv6 datagrams with header errors (length, version mismatch). The value in the IPv6 headers Length field is used to update this counter	RO	0x0

gmacgrp_rxipv6_nopay_octets

Number of bytes received in IPv6 datagrams that did not have a TCP, UDP, or ICMP payload. The value in the IPv6 headers Length field is used to update this counter

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF80026C
i_emac_emac1	0xFF802000	0xFF80226C
i_emac_emac2	0xFF804000	0xFF80426C

Offset: 0x26C

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_rxipv6_nopay_octets Fields

Bit	Name	Description	Access	Reset
31:0	cnt	Number of bytes received in IPv6 datagrams that did not have a TCP, UDP, or ICMP payload. The value in the IPv6 headers Length field is used to update this counter	RO	0x0

gmacgrp_rxudp_gd_octets

Number of bytes received in a good UDP segment. This counter does not count IP header bytes

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800270
i_emac_emac1	0xFF802000	0xFF802270
i_emac_emac2	0xFF804000	0xFF804270

Offset: 0x270

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_rxudp_gd_octets Fields

Bit	Name	Description	Access	Reset
31:0	cnt	Number of bytes received in a good UDP segment. This counter does not count IP header bytes	RO	0x0

gmacgrp_rxudp_err_octets

Number of bytes received in a UDP segment that had checksum errors

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800274
i_emac_emac1	0xFF802000	0xFF802274
i_emac_emac2	0xFF804000	0xFF804274

Offset: 0x274

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_rxudp_err_octets Fields

Bit	Name	Description	Access	Reset
31:0	cnt	Number of bytes received in a UDP segment that had checksum errors	RO	0x0

gmacgrp_rxtcp_gd_octets

Number of bytes received in a good TCP segment

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800278
i_emac_emac1	0xFF802000	0xFF802278
i_emac_emac2	0xFF804000	0xFF804278

Offset: 0x278

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_rxtcp_gd_octets Fields

Bit	Name	Description	Access	Reset
31:0	cnt	Number of bytes received in a good TCP segment	RO	0x0

gmacgrp_rxtcperroctets

Number of bytes received in a TCP segment with checksum errors

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF80027C
i_emac_emac1	0xFF802000	0xFF80227C
i_emac_emac2	0xFF804000	0xFF80427C

Offset: 0x27C

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
rxtcp_err_octets RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rxtcp_err_octets RO 0x0															

gmacgrp_rxtcperroctets Fields

Bit	Name	Description	Access	Reset
31:0	rxtcp_err_octets	Number of bytes received in a TCP segment with checksum errors	RO	0x0

gmacgrp_rxicmp_gd_octets

Number of bytes received in a good ICMP segment

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800280
i_emac_emac1	0xFF802000	0xFF802280
i_emac_emac2	0xFF804000	0xFF804280

Offset: 0x280

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_rxicmp_gd_octets Fields

Bit	Name	Description	Access	Reset
31:0	cnt	Number of bytes received in a good ICMP segment	RO	0x0

gmacgrp_rxicmp_err_octets

Number of bytes received in an ICMP segment with checksum errors

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800284
i_emac_emac1	0xFF802000	0xFF802284
i_emac_emac2	0xFF804000	0xFF804284

Offset: 0x284

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cnt RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cnt RO 0x0															

gmacgrp_rxicmp_err_octets Fields

Bit	Name	Description	Access	Reset
31:0	cnt	Number of bytes received in an ICMP segment with checksum errors	RO	0x0

gmacgrp_l3_l4_control0

Register 256 (Layer 3 and Layer 4 Control Register 0)

This register controls the operations of the filter 0 of Layer 3 and Layer 4.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800400
i_emac_emac1	0xFF802000	0xFF802400
i_emac_emac2	0xFF804000	0xFF804400

Offset: 0x400

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
reserved_31_22 RO 0x0										l4dpim0 RW 0x0	l4dpm0 RW 0x0	l4spim0 RW 0x0	l4spm0 RW 0x0	reserved_17 RO 0x0	l4pen0 RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
l3hdbm0 RW 0x0					l3hsbm0 RW 0x0					l3daim0 RW 0x0	l3dam0 RW 0x0	l3saim0 RW 0x0	l3sam0 RW 0x0	reserved_1 RO 0x0	l3pen0 RW 0x0

gmacgrp_l3_l4_control0 Fields

Bit	Name	Description	Access	Reset
31:22	reserved_31_22	Reserved	RO	0x0
21	l4dpim0	Layer 4 Destination Port Inverse Match Enable When set, this bit indicates that the Layer 4 Destination Port number field is enabled for inverse matching. When reset, this bit indicates that the Layer 4 Destination Port number field is enabled for perfect matching. This bit is valid and applicable only when Bit 20 (L4DPM0) is set high.	RW	0x0
20	l4dpm0	Layer 4 Destination Port Match Enable When set, this bit indicates that the Layer 4 Destination Port number field is enabled for matching. When reset, the MAC ignores the Layer 4 Destination Port number field for matching.	RW	0x0

Bit	Name	Description	Access	Reset
19	l4spim0	Layer 4 Source Port Inverse Match Enable When set, this bit indicates that the Layer 4 Source Port number field is enabled for inverse matching. When reset, this bit indicates that the Layer 4 Source Port number field is enabled for perfect matching. This bit is valid and applicable only when Bit 18 (L4SPM0) is set high.	RW	0x0
18	l4spm0	Layer 4 Source Port Match Enable When set, this bit indicates that the Layer 4 Source Port number field is enabled for matching. When reset, the MAC ignores the Layer 4 Source Port number field for matching.	RW	0x0
17	reserved_17	Reserved	RO	0x0
16	l4pen0	Layer 4 Protocol Enable When set, this bit indicates that the Source and Destination Port number fields for UDP frames are used for matching. When reset, this bit indicates that the Source and Destination Port number fields for TCP frames are used for matching. The Layer 4 matching is done only when either L4SPM0 or L4DPM0 bit is set high.	RW	0x0

Bit	Name	Description	Access	Reset
15:11	l3hdbm0	Layer 3 IP DA Higher Bits Match IPv4 Frames: This field contains the number of higher bits of IP Destination Address that are matched in the IPv4 frames. The following list describes the values of this field: * 0: No bits are masked. * 1: LSb[0] is masked. * 2: Two LSbs [1:0] are masked. * ... * 31: All bits except MSb are masked. IPv6 Frames: Bits [12:11] of this field correspond to Bits [6:5] of L3HSBM0, which indicate the number of lower bits of IP Source or Destination Address that are masked in the IPv6 frames. The following list describes the concatenated values of the L3HDBM0[1:0] and L3HSBM0 bits: * 0: No bits are masked. * 1: LSb[0] is masked. * 2: Two LSbs [1:0] are masked. * ... * 127: All bits except MSb are masked. This field is valid and applicable only if L3DAM0 or L3SAM0 is set high.	RW	0x0

Bit	Name	Description	Access	Reset
10:6	l3hsbm0	Layer 3 IP SA Higher Bits Match IPv4 Frames: This field contains the number of lower bits of IP Source Address that are masked for matching in the IPv4 frames. The following list describes the values of this field: * 0: No bits are masked. * 1: LSb[0] is masked. * 2: Two LSbs [1:0] are masked. * ... * 31: All bits except MSb are masked. IPv6 Frames: This field contains Bits [4:0] of the field that indicates the number of higher bits of IP Source or Destination Address matched in the IPv6 frames. This field is valid and applicable only if L3DAM0 or L3SAM0 is set high.	RW	0x0
5	l3daim0	Layer 3 IP DA Inverse Match Enable When set, this bit indicates that the Layer 3 IP Destination Address field is enabled for inverse matching. When reset, this bit indicates that the Layer 3 IP Destination Address field is enabled for perfect matching. This bit is valid and applicable only when Bit 4 (L3DAM0) is set high.	RW	0x0

Bit	Name	Description	Access	Reset
4	l3dam0	Layer 3 IP DA Match Enable When set, this bit indicates that Layer 3 IP Destination Address field is enabled for matching. When reset, the MAC ignores the Layer 3 IP Destination Address field for matching. Note: When Bit 0 (L3PEN0) is set, you should set either this bit or Bit 2 (L3SAM0) because either IPv6 DA or SA can be checked for filtering.	RW	0x0
3	l3saim0	Layer 3 IP SA Inverse Match Enable When set, this bit indicates that the Layer 3 IP Source Address field is enabled for inverse matching. When reset, this bit indicates that the Layer 3 IP Source Address field is enabled for perfect matching. This bit is valid and applicable only when Bit 2 (L3SAM0) is set high.	RW	0x0
2	l3sam0	Layer 3 IP SA Match Enable When set, this bit indicates that the Layer 3 IP Source Address field is enabled for matching. When reset, the MAC ignores the Layer 3 IP Source Address field for matching. Note: When Bit 0 (L3PEN0) is set, you should set either this bit or Bit 4 (L3DAM0) because either IPv6 SA or DA can be checked for filtering.	RW	0x0
1	reserved_1	Reserved	RO	0x0

Bit	Name	Description	Access	Reset
0	l3pen0	Layer 3 Protocol Enable When set, this bit indicates that the Layer 3 IP Source or Destination Address matching is enabled for the IPv6 frames. When reset, this bit indicates that the Layer 3 IP Source or Destination Address matching is enabled for the IPv4 frames. The Layer 3 matching is done only when either L3SAM0 or L3DAM0 bit is set high.	RW	0x0

gmacgrp_layer4_address0

Register 257 (Layer 4 Address Register 0)

You can configure the Layer 3 and Layer 4 Address Registers to be double-synchronized by selecting the Synchronize Layer 3 and Layer 4 Address Registers to Rx Clock Domain option in coreConsultant. If the Layer 3 and Layer 4 Address Registers are configured to be double-synchronized to the Rx clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the Layer 3 and Layer 4 Address Registers are written. For proper synchronization updates, you should perform the consecutive writes to the same Layer 3 and Layer 4 Address Registers after at least four clock cycles delay of the destination clock.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800404
i_emac_emac1	0xFF802000	0xFF802404
i_emac_emac2	0xFF804000	0xFF804404

Offset: 0x404

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
l4dp0 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
l4sp0 RW 0x0															

gmacgrp_layer4_address0 Fields

Bit	Name	Description	Access	Reset
31:16	l4dp0	Layer 4 Destination Port Number Field When Bit 16 (L4PEN0) is reset and Bit 20 (L4DPM0) is set in Register 256 (Layer 3 and Layer 4 Control Register 0), this field contains the value to be matched with the TCP Destination Port Number field in the IPv4 or IPv6 frames. When Bit 16 (L4PEN0) and Bit 20 (L4DPM0) are set in Register 256 (Layer 3 and Layer 4 Control Register 0), this field contains the value to be matched with the UDP Destination Port Number field in the IPv4 or IPv6 frames.	RW	0x0
15:0	l4sp0	Layer 4 Source Port Number Field Layer 4 Source Port Number Field When Bit 16 (L4PEN0) is reset and Bit 20 (L4DPM0) is set in Register 256 (Layer 3 and Layer 4 Control Register 0), this field contains the value to be matched with the TCP Source Port Number field in the IPv4 or IPv6 frames. When Bit 16 (L4PEN0) and Bit 20 (L4DPM0) are set in Register 256 (Layer 3 and Layer 4 Control Register 0), this field contains the value to be matched with the UDP Source Port Number field in the IPv4 or IPv6 frames.	RW	0x0

gmacgrp_layer3_addr0_reg0

Register 260 (Layer 3 Address 0 Register 0)

For IPv4 frames, the Layer 3 Address 0 Register 0 contains the 32-bit IP Source Address field. For IPv6 frames, it contains Bits[31:0] of the 128-bit IP Source Address or Destination Address field.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800410
i_emac_emac1	0xFF802000	0xFF802410
i_emac_emac2	0xFF804000	0xFF804410

Offset: 0x410

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
l3a00 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
l3a00 RW 0x0															

gmacgrp_layer3_addr0_reg0 Fields

Bit	Name	Description	Access	Reset
31:0	l3a00	Layer 3 Address 0 Field When Bit 0 (L3PEN0) and Bit 2 (L3SAM0) are set in Register 256 (Layer 3 and Layer 4 Control Register 0), this field contains the value to be matched with Bits[31:0] of the IP Source Address field in the IPv6 frames. When Bit 0 (L3PEN0) and Bit 4 (L3DAM0) are set in Register 256 (Layer 3 and Layer 4 Control Register 0), this field contains the value to be matched with Bits [31:0] of the IP Destination Address field in the IPv6 frames. When Bit 0 (L3PEN0) is reset and Bit 2 (L3SAM0) is set in Register 256 (Layer 3 and Layer 4 Control Register 0), this field contains the value to be matched with the IP Source Address field in the IPv4 frames.	RW	0x0

gmacgrp_layer3_addr1_reg0

Register 261 (Layer 3 Address 1 Register 0)

For IPv4 frames, the Layer 3 Address 1 Register 0 contains the 32-bit IP Destination Address field. For IPv6 frames, it contains Bits[63:32] of the 128-bit IP Source Address or Destination Address field.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800414
i_emac_emac1	0xFF802000	0xFF802414
i_emac_emac2	0xFF804000	0xFF804414

Offset: 0x414

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
l3a10 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
l3a10 RW 0x0															

gmacgrp_layer3_addr1_reg0 Fields

Bit	Name	Description	Access	Reset
31:0	l3a10	Layer 3 Address 1 Field When Bit 0 (L3PEN0) and Bit 2 (L3SAM0) are set in Register 256 (Layer 3 and Layer 4 Control Register 0), this field contains the value to be matched with Bits [63:32] of the IP Source Address field in the IPv6 frames. When Bit 0 (L3PEN0) and Bit 4 (L3DAM0) are set in Register 256 (Layer 3 and Layer 4 Control Register 0), this field contains the value to be matched with Bits [63:32] of the IP Destination Address field in the IPv6 frames. When Bit 0 (L3PEN0) is reset and Bit 4 (L3DAM0) is set in Register 256 (Layer 3 and Layer 4 Control Register 0), this field contains the value to be matched with the IP Destination Address field in the IPv4 frames.	RW	0x0

gmacgrp_layer3_addr2_reg0

Register 262 (Layer 3 Address 2 Register 0)

For IPv4 frames, the Layer 3 Address 2 Register 0 is reserved. For IPv6 frames, it contains Bits [95:64] of the 128-bit IP Source Address or Destination Address field.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800418
i_emac_emac1	0xFF802000	0xFF802418
i_emac_emac2	0xFF804000	0xFF804418

Offset: 0x418

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
l3a20 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
l3a20 RW 0x0															

gmacgrp_layer3_addr2_reg0 Fields

Bit	Name	Description	Access	Reset
31:0	l3a20	Layer 3 Address 2 Field When Bit 0 (L3PEN0) and Bit 2 (L3SAM0) are set in Register 256 (Layer 3 and Layer 4 Control Register 0), this field contains the value to be matched with Bits [95:64] of the IP Source Address field in the IPv6 frames. When Bit 0 (L3PEN0) and Bit 4 (L3DAM0) are set in Register 256 (Layer 3 and Layer 4 Control Register 0), this field contains value to be matched with Bits [95:64] of the IP Destination Address field in the IPv6 frames. When Bit 0 (L3PEN0) is reset in Register 256 (Layer 3 and Layer 4 Control Register 0), this register is not used.	RW	0x0

gmacgrp_layer3_addr3_reg0

Register 263 (Layer 3 Address 3 Register 0)

For IPv4 frames, the Layer 3 Address 3 Register 0 is reserved. For IPv6 frames, it contains Bits [127:96] of the 128-bit IP Source Address or Destination Address field.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF80041C
i_emac_emac1	0xFF802000	0xFF80241C
i_emac_emac2	0xFF804000	0xFF80441C

Offset: 0x41C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
l3a30 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
l3a30 RW 0x0															

gmacgrp_layer3_addr3_reg0 Fields

Bit	Name	Description	Access	Reset
31:0	l3a30	Layer 3 Address 3 Field When Bit 0 (L3PEN0) and Bit 2 (L3SAM0) are set in Register 256 (Layer 3 and Layer 4 Control Register 0), this field contains the value to be matched with Bits [127:96] of the IP Source Address field in the IPv6 frames. When Bit 0 (L3PEN0) and Bit 4 (L3DAM0) are set in Register 256 (Layer 3 and Layer 4 Control Register 0), this field contains the value to be matched with Bits [127:96] of the IP Destination Address field in the IPv6 frames. When Bit 0 (L3PEN0) is reset in Register 256 (Layer 3 and Layer 4 Control Register 0), this register is not used.	RW	0x0

gmacgrp_l3_l4_control1

This register controls the operations of the filter 0 of Layer 3 and Layer 4.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800430
i_emac_emac1	0xFF802000	0xFF802430
i_emac_emac2	0xFF804000	0xFF804430

Offset: 0x430

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved										l4dpim1 RW 0x0	l4dpm1 RW 0x0	l4spim1 RW 0x0	l4spm1 RW 0x0	Reserved	l4pen1 RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
l3hdbm1 RW 0x0					l3hsbm1 RW 0x0					l3daim1 RW 0x0	l3dam1 RW 0x0	l3saim1 RW 0x0	l3sam1 RW 0x0	Reserved	l3pen1 RW 0x0

gmacgrp_l3_l4_control1 Fields

Bit	Name	Description	Access	Reset
21	l4dpim1	When set, this bit indicates that the Layer 4 Destination Port number field is enabled for inverse matching. When reset, this bit indicates that the Layer 4 Destination Port number field is enabled for perfect matching. This bit is valid and applicable only when Bit 20 (L4DPM1) is set high.	RW	0x0
20	l4dpm1	When set, this bit indicates that the Layer 4 Destination Port number field is enabled for matching. When reset, the MAC ignores the Layer 4 Destination Port number field for matching.	RW	0x0
19	l4spim1	When set, this bit indicates that the Layer 4 Source Port number field is enabled for inverse matching. When reset, this bit indicates that the Layer 4 Source Port number field is enabled for perfect matching. This bit is valid and applicable only when Bit 18 (L4SPM1) is set high.	RW	0x0

Bit	Name	Description	Access	Reset
18	l4spm1	When set, this bit indicates that the Layer 4 Source Port number field is enabled for matching. When reset, the MAC ignores the Layer 4 Source Port number field for matching.	RW	0x0
16	l4pen1	When set, this bit indicates that the Source and Destination Port number fields for UDP frames are used for matching. When reset, this bit indicates that the Source and Destination Port number fields for TCP frames are used for matching. The Layer 4 matching is done only when either L4SPM1 or L4DPM1 bit is set high.	RW	0x0
15:11	l3hdbm1	IPv4 Frames: This field contains the number of higher bits of IP Destination Address that are matched in the IPv4 frames. The following list describes the values of this field: * 0: No bits are masked. * 1: LSb[0] is masked. * 2: Two LSbs [1:0] are masked. * ... * 31: All bits except MSb are masked. IPv6 Frames: Bits [12:11] of this field correspond to Bits [6:5] of L3HSBM1, which indicate the number of lower bits of IP Source or Destination Address that are masked in the IPv6 frames. The following list describes the concatenated values of the L3HDBM1[1:0] and L3HSBM1 bits: * 0: No bits are masked. * 1: LSb[0] is masked. * 2: Two LSbs [1:0] are masked. * ... * 127: All bits except MSb are masked. This field is valid and applicable only if L3DAM1 or L3SAM1 is set high.	RW	0x0

Bit	Name	Description	Access	Reset
10:6	l3hsbm1	IPv4 Frames: This field contains the number of lower bits of IP Source Address that are masked for matching in the IPv4 frames. The following list describes the values of this field: * 0: No bits are masked. * 1: LSb[0] is masked. * 2: Two LSbs [1:0] are masked. * ... * 31: All bits except MSb are masked. IPv6 Frames: This field contains Bits [4:0] of the field that indicates the number of higher bits of IP Source or Destination Address matched in the IPv6 frames. This field is valid and applicable only if L3DAM1 or L3SAM1 is set high.	RW	0x0
5	l3daim1	When set, this bit indicates that the Layer 3 IP Destination Address field is enabled for inverse matching. When reset, this bit indicates that the Layer 3 IP Destination Address field is enabled for perfect matching. This bit is valid and applicable only when Bit 4 (L3DAM1) is set high.	RW	0x0
4	l3dam1	When set, this bit indicates that Layer 3 IP Destination Address field is enabled for matching. When reset, the MAC ignores the Layer 3 IP Destination Address field for matching. Note: When Bit 1 (L3PEN1) is set, you should set either this bit or Bit 2 (L3SAM1) because either IPv6 DA or SA can be checked for filtering.	RW	0x0

Bit	Name	Description	Access	Reset
3	l3saim1	When set, this bit indicates that the Layer 3 IP Source Address field is enabled for inverse matching. When reset, this bit indicates that the Layer 3 IP Source Address field is enabled for perfect matching. This bit is valid and applicable only when Bit 2 (L3SAM1) is set high.	RW	0x0
2	l3sam1	When set, this bit indicates that the Layer 3 IP Source Address field is enabled for matching. When reset, the MAC ignores the Layer 3 IP Source Address field for matching. Note: When Bit 0 (L3PEN1) is set, you should set either this bit or Bit 4 (L3DAM1) because either IPv6 SA or DA can be checked for filtering.	RW	0x0
0	l3pen1	When set, this bit indicates that the Layer 3 IP Source or Destination Address matching is enabled for the IPv6 frames. When reset, this bit indicates that the Layer 3 IP Source or Destination Address matching is enabled for the IPv4 frames. The Layer 3 matching is done only when either L3SAM1 or L3DAM1 bit is set high.	RW	0x0

gmacgrp_layer4_address1

Because the Layer 3 and Layer 4 Address Registers are double-synchronized to the Rx clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the Layer 3 and Layer 4 Address Registers are written. For proper synchronization updates, you should perform the consecutive writes to the same Layer 3 and Layer 4 Address Registers after at least four clock cycles delay of the destination clock.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800434
i_emac_emac1	0xFF802000	0xFF802434
i_emac_emac2	0xFF804000	0xFF804434

Offset: 0x434

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
l4dp1 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
l4sp1 RW 0x0															

gmacgrp_layer4_address1 Fields

Bit	Name	Description	Access	Reset
31:16	l4dp1	When Bit 16 (L4PEN1) is reset and Bit 20 (L4DPM1) is set in Register 268 (Layer 3 and Layer 4 Control Register 0), this field contains the value to be matched with the TCP Destination Port Number field in the IPv4 or IPv6 frames. When Bit 16 (L4PEN1) and Bit 20 (L4DPM1) are set in Register 268 (Layer 3 and Layer 4 Control Register 1), this field contains the value to be matched with the UDP Destination Port Number field in the IPv4 or IPv6 frames.	RW	0x0

Bit	Name	Description	Access	Reset
15:0	l4sp1	When Bit 16 (L4PEN1) is reset and Bit 20 (L4DPM1) is set in Register 268 (Layer 3 and Layer 4 Control Register 1), this field contains the value to be matched with the TCP Source Port Number field in the IPv4 or IPv6 frames. When Bit 16 (L4PEN1) and Bit 20 (L4DPM1) are set in Register 268 (Layer 3 and Layer 4 Control Register 1), this field contains the value to be matched with the UDP Source Port Number field in the IPv4 or IPv6 frames.	RW	0x0

gmacgrp_layer3_addr0_reg1

For IPv4 frames, the Layer 3 Address 0 Register 1 contains the 32-bit IP Source Address field. For IPv6 frames, it contains Bits[31:0] of the 128-bit IP Source Address or Destination Address field.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800440
i_emac_emac1	0xFF802000	0xFF802440
i_emac_emac2	0xFF804000	0xFF804440

Offset: 0x440

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
l3a01 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
l3a01 RW 0x0															

gmacgrp_layer3_addr0_reg1 Fields

Bit	Name	Description	Access	Reset
31:0	l3a01	When Bit 0 (L3PEN1) and Bit 2 (L3SAM1) are set in Register 268 (Layer 3 and Layer 4 Control Register 1), this field contains the value to be matched with Bits[31:0] of the IP Source Address field in the IPv6 frames. When Bit 0 (L3PEN1) and Bit 4 (L3DAM1) are set in Register 268 (Layer 3 and Layer 4 Control Register 1), this field contains the value to be matched with Bits [31:0] of the IP Destination Address field in the IPv6 frames. When Bit 0 (L3PEN1) is reset and Bit 2 (L3SAM1) is set in Register 268 (Layer 3 and Layer 4 Control Register 1), this field contains the value to be matched with the IP Source Address field in the IPv4 frames.	RW	0x0

gmacgrp_layer3_addr1_reg1

For IPv4 frames, the Layer 3 Address 1 Register 1 contains the 32-bit IP Destination Address field. For IPv6 frames, it contains Bits[63:32] of the 128-bit IP Source Address or Destination Address field

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800444
i_emac_emac1	0xFF802000	0xFF802444
i_emac_emac2	0xFF804000	0xFF804444

Offset: 0x444

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
l3a11 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
l3a11 RW 0x0															

gmacgrp_layer3_addr1_reg1 Fields

Bit	Name	Description	Access	Reset
31:0	l3a11	When Bit 0 (L3PEN1) and Bit 2 (L3SAM1) are set in Register 268 (Layer 3 and Layer 4 Control Register 1), this field contains the value to be matched with Bits [63:32] of the IP Source Address field in the IPv6 frames. When Bit 0 (L3PEN1) and Bit 4 (L3DAM1) are set in Register 268 (Layer 3 and Layer 4 Control Register 1), this field contains the value to be matched with Bits [63:32] of the IP Destination Address field in the IPv6 frames. When Bit 0 (L3PEN1) is reset and Bit 4 (L3DAM1) is set in Register 268 (Layer 3 and Layer 4 Control Register 1), this field contains the value to be matched with the IP Destination Address field in the IPv4 frames.	RW	0x0

gmacgrp_layer3_addr2_reg1

For IPv4 frames, the Layer 3 Address 2 Register 1 is reserved. For IPv6 frames, it contains Bits [95:64] of the 128-bit IP Source Address or Destination Address field.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800448
i_emac_emac1	0xFF802000	0xFF802448

Module Instance	Base Address	Register Address
i_emac_emac2	0xFF804000	0xFF804448

Offset: 0x448

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
l3a21 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
l3a21 RW 0x0															

gmacgrp_layer3_addr2_reg1 Fields

Bit	Name	Description	Access	Reset
31:0	l3a21	When Bit 0 (L3PEN1) and Bit 2 (L3SAM1) are set in Register 268 (Layer 3 and Layer 4 Control Register 1), this field contains the value to be matched with Bits [95:64] of the IP Source Address field in the IPv6 frames. When Bit 0 (L3PEN1) and Bit 4 (L3DAM1) are set in Register 268 (Layer 3 and Layer 4 Control Register 1), this field contains value to be matched with Bits [95:64] of the IP Destination Address field in the IPv6 frames.	RW	0x0

gmacgrp_layer3_addr3_reg1

For IPv4 frames, the Layer 3 Address 3 Register 1 is reserved. For IPv6 frames, it contains Bits [127:96] of the 128-bit IP Source Address or Destination Address field.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF80044C

Module Instance	Base Address	Register Address
i_emac_emac1	0xFF802000	0xFF80244C
i_emac_emac2	0xFF804000	0xFF80444C

Offset: 0x44C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
l3a31 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
l3a31 RW 0x0															

gmacgrp_layer3_addr3_reg1 Fields

Bit	Name	Description	Access	Reset
31:0	l3a31	When Bit 1 (L3PEN1) and Bit 2 (L3SAM1) are set in Register 268 (Layer 3 and Layer 4 Control Register 1), this field contains the value to be matched with Bits [127:96] of the IP Source Address field in the IPv6 frames. When Bit 0 (L3PEN1) and Bit 4 (L3DAM1) are set in Register 268 (Layer 3 and Layer 4 Control Register 1), this field contains the value to be matched with Bits [127:96] of the IP Destination Address field in the IPv6 frames. When Bit 0 (L3PEN1) is reset in Register 268 (Layer 3 and Layer 4 Control Register 1), this register is not used.	RW	0x0

gmacgrp_l3_l4_control2

This register controls the operations of the filter 2 of Layer 3 and Layer 4.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800460
i_emac_emac1	0xFF802000	0xFF802460
i_emac_emac2	0xFF804000	0xFF804460

Offset: 0x460

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved										l4dpim2 RW 0x0	l4dpm2 RW 0x0	l4spim2 RW 0x0	l4spm2 RW 0x0	Reserved	l4pen2 RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
l3hdbm2 RW 0x0					l3hsbm2 RW 0x0					l3daim2 RW 0x0	l3dam2 RW 0x0	l3saim2 RW 0x0	l3sam2 RW 0x0	Reserved	l3pen2 RW 0x0

gmacgrp_l3_l4_control2 Fields

Bit	Name	Description	Access	Reset
21	l4dpim2	When set, this bit indicates that the Layer 4 Destination Port number field is enabled for inverse matching. When reset, this bit indicates that the Layer 4 Destination Port number field is enabled for perfect matching. This bit is valid and applicable only when Bit 20 (L4DPM0) is set high.	RW	0x0
20	l4dpm2	When set, this bit indicates that the Layer 4 Destination Port number field is enabled for matching. When reset, the MAC ignores the Layer 4 Destination Port number field for matching.	RW	0x0

Bit	Name	Description	Access	Reset
19	l4spim2	When set, this bit indicates that the Layer 4 Source Port number field is enabled for inverse matching. When reset, this bit indicates that the Layer 4 Source Port number field is enabled for perfect matching. This bit is valid and applicable only when Bit 18 (L4SPM2) is set high.	RW	0x0
18	l4spm2	When set, this bit indicates that the Layer 4 Source Port number field is enabled for matching. When reset, the MAC ignores the Layer 4 Source Port number field for matching.	RW	0x0
16	l4pen2	When set, this bit indicates that the Source and Destination Port number fields for UDP frames are used for matching. When reset, this bit indicates that the Source and Destination Port number fields for TCP frames are used for matching. The Layer 4 matching is done only when either L4SPM2 or L4DPM2 bit is set high.	RW	0x0

Bit	Name	Description	Access	Reset
15:11	l3hdbm2	IPv4 Frames: This field contains the number of higher bits of IP Destination Address that are matched in the IPv4 frames. The following list describes the values of this field: * 0: No bits are masked. * 1: LSb[0] is masked. * 2: Two LSbs [1:0] are masked. * ... * 31: All bits except MSb are masked. IPv6 Frames: Bits [12:11] of this field correspond to Bits [6:5] of L3HSBM2, which indicate the number of lower bits of IP Source or Destination Address that are masked in the IPv6 frames. The following list describes the concatenated values of the L3HDBM2[1:0] and L3HSBM2 bits: * 0: No bits are masked. * 1: LSb[0] is masked. * 2: Two LSbs [1:0] are masked. * ... * 127: All bits except MSb are masked. This field is valid and applicable only if L3DAM2 or L3SAM2 is set high.	RW	0x0

Bit	Name	Description	Access	Reset
10:6	l3hsbm2	Layer 3 IP SA Higher Bits Match IPv4 Frames: This field contains the number of lower bits of IP Source Address that are masked for matching in the IPv4 frames. The following list describes the values of this field: * 0: No bits are masked. * 1: LSb[0] is masked. * 2: Two LSbs [1:0] are masked. * ... * 31: All bits except MSb are masked. IPv6 Frames: This field contains Bits [4:0] of the field that indicates the number of higher bits of IP Source or Destination Address matched in the IPv6 frames. This field is valid and applicable only if L3DAM2 or L3SAM2 is set high.	RW	0x0
5	l3daim2	When set, this bit indicates that the Layer 3 IP Destination Address field is enabled for inverse matching. When reset, this bit indicates that the Layer 3 IP Destination Address field is enabled for perfect matching. This bit is valid and applicable only when Bit 4 (L3DAM2) is set high.	RW	0x0
4	l3dam2	When set, this bit indicates that Layer 3 IP Destination Address field is enabled for matching. When reset, the MAC ignores the Layer 3 IP Destination Address field for matching. Note: When Bit 0 (L3PEN2) is set, you should set either this bit or Bit 2 (L3SAM2) because either IPv6 DA or SA can be checked for filtering.	RW	0x0

Bit	Name	Description	Access	Reset
3	l3saim2	When set, this bit indicates that the Layer 3 IP Source Address field is enabled for inverse matching. When reset, this bit indicates that the Layer 3 IP Source Address field is enabled for perfect matching. This bit is valid and applicable only when Bit 2 (L3SAM2) is set high.	RW	0x0
2	l3sam2	When set, this bit indicates that the Layer 3 IP Source Address field is enabled for matching. When reset, the MAC ignores the Layer 3 IP Source Address field for matching. Note: When Bit 0 (L3PEN2) is set, you should set either this bit or Bit 4 (L3DAM2) because either IPv6 SA or DA can be checked for filtering.	RW	0x0
0	l3pen2	When set, this bit indicates that the Layer 3 IP Source or Destination Address matching is enabled for the IPv6 frames. When reset, this bit indicates that the Layer 3 IP Source or Destination Address matching is enabled for the IPv4 frames. The Layer 3 matching is done only when either L3SAM2 or L3DAM2 bit is set high.	RW	0x0

gmacgrp_layer4_address2

Because the Layer 3 and Layer 4 Address Registers are double-synchronized to the Rx clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the Layer 3 and Layer 4 Address Registers are written. For proper synchronization updates, you should perform the consecutive writes to the same Layer 3 and Layer 4 Address Registers after at least four clock cycles delay of the destination clock.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800464
i_emac_emac1	0xFF802000	0xFF802464
i_emac_emac2	0xFF804000	0xFF804464

Offset: 0x464

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
l4dp2 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
l4sp2 RW 0x0															

gmacgrp_layer4_address2 Fields

Bit	Name	Description	Access	Reset
31:16	l4dp2	When Bit 16 (L4PEN2) is reset and Bit 20 (L4DPM2) is set in Register 280 (Layer 3 and Layer 4 Control Register 2), this field contains the value to be matched with the TCP Destination Port Number field in the IPv4 or IPv6 frames. When Bit 16 (L4PEN2) and Bit 20 (L4DPM2) are set in Register 280 (Layer 3 and Layer 4 Control Register 2), this field contains the value to be matched with the UDP Destination Port Number field in the IPv4 or IPv6 frames.	RW	0x0

Bit	Name	Description	Access	Reset
15:0	l4sp2	When Bit 16 (L4PEN2) is reset and Bit 20 (L4DPM2) is set in Register 280 (Layer 3 and Layer 4 Control Register 2), this field contains the value to be matched with the TCP Source Port Number field in the IPv4 or IPv6 frames. When Bit 16 (L4PEN2) and Bit 20 (L4DPM2) are set in Register 280 (Layer 3 and Layer 4 Control Register 2), this field contains the value to be matched with the UDP Source Port Number field in the IPv4 or IPv6 frames.	RW	0x0

gmacgrp_layer3_addr0_reg2

For IPv4 frames, the Layer 3 Address 0 Register 2 contains the 32-bit IP Source Address field. For IPv6 frames, it contains Bits [31:0] of the 128-bit IP Source Address or Destination Address field.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800470
i_emac_emac1	0xFF802000	0xFF802470
i_emac_emac2	0xFF804000	0xFF804470

Offset: 0x470

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
l3a02 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
l3a02 RW 0x0															

gmacgrp_layer3_addr0_reg2 Fields

Bit	Name	Description	Access	Reset
31:0	l3a02	When Bit 0 (L3PEN2) and Bit 2 (L3SAM2) are set in Register 280 (Layer 3 and Layer 4 Control Register 2), this field contains the value to be matched with Bits [31:0] of the IP Source Address field in the IPv6 frames. When Bit 0 (L3PEN2) and Bit 4 (L3DAM2) are set in Register 280 (Layer 3 and Layer 4 Control Register 2), this field contains the value to be matched with Bits [31:0] of the IP Destination Address field in the IPv6 frames. When Bit 0 (L3PEN2) is reset and Bit 2 (L3SAM2) is set in Register 280 (Layer 3 and Layer 4 Control Register 2), this field contains the value to be matched with the IP Source Address field in the IPv4 frames.	RW	0x0

gmacgrp_layer3_addr1_reg2

For IPv4 frames, the Layer 3 Address 1 Register 2 contains the 32-bit IP Destination Address field. For IPv6 frames, it contains Bits [63:32] of the 128-bit IP Source Address or Destination Address field.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800474
i_emac_emac1	0xFF802000	0xFF802474
i_emac_emac2	0xFF804000	0xFF804474

Offset: 0x474

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
l3a12 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
l3a12 RW 0x0															

gmacgrp_layer3_addr1_reg2 Fields

Bit	Name	Description	Access	Reset
31:0	l3a12	Layer 3 Address 1 Field When Bit 0 (L3PEN2) and Bit 2 (L3SAM2) are set in Register 280 (Layer 3 and Layer 4 Control Register 2), this field contains the value to be matched with Bits [63:32] of the IP Source Address field in the IPv6 frames. When Bit 0 (L3PEN2) and Bit 4 (L3DAM2) are set in Register 280 (Layer 3 and Layer 4 Control Register 2), this field contains the value to be matched with Bits [63:32] of the IP Destination Address field in the IPv6 frames. When Bit 0 (L3PEN2) is reset and Bit 4 (L3DAM2) is set in Register 280 (Layer 3 and Layer 4 Control Register 2), this field contains the value to be matched with the IP Destination Address field in the IPv4 frames.	RW	0x0

gmacgrp_layer3_addr2_reg2

For IPv4 frames, the Layer 3 Address 2 Register 2 is reserved. For IPv6 frames, it contains Bits [95:64] of the 128-bit IP Source Address or Destination Address field.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800478
i_emac_emac1	0xFF802000	0xFF802478
i_emac_emac2	0xFF804000	0xFF804478

Offset: 0x478

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
l3a22 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
l3a22 RW 0x0															

gmacgrp_layer3_addr2_reg2 Fields

Bit	Name	Description	Access	Reset
31:0	l3a22	When Bit 0 (L3PEN2) and Bit 2 (L3SAM2) are set in Register 280 (Layer 3 and Layer 4 Control Register 2), this field contains the value to be matched with Bits [95:64] of the IP Source Address field in the IPv6 frames. When Bit 0 (L3PEN2) and Bit 4 (L3DAM2) are set in Register 256 (Layer 3 and Layer 4 Control Register 2), this field contains value to be matched with Bits [95:64] of the IP Destination Address field in the IPv6 frames. When Bit 0 (L3PEN2) is reset in Register 280 (Layer 3 and Layer 4 Control Register 2), this register is not used.	RW	0x0

gmacgrp_layer3_addr3_reg2

For IPv4 frames, the Layer 3 Address 3 Register 2 is reserved. For

IPv6 frames, it contains Bits [127:96] of the 128-bit IP Source Address or Destination Address field.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF80047C
i_emac_emac1	0xFF802000	0xFF80247C
i_emac_emac2	0xFF804000	0xFF80447C

Offset: 0x47C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
l3a32 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
l3a32 RW 0x0															

gmacgrp_layer3_addr3_reg2 Fields

Bit	Name	Description	Access	Reset
31:0	l3a32	When Bit 0 (L3PEN2) and Bit 2 (L3SAM2) are set in Register 280 (Layer 3 and Layer 4 Control Register 2), this field contains the value to be matched with Bits [127:96] of the IP Source Address field in the IPv6 frames. When Bit 0 (L3PEN2) and Bit 4 (L3DAM2) are set in Register 280 (Layer 3 and Layer 4 Control Register 2), this field contains the value to be matched with Bits [127:96] of the IP Destination Address field in the IPv6 frames. When Bit 0 (L3PEN2) is reset in Register 280 (Layer 3 and Layer 4 Control Register 2), this register is not used.	RW	0x0

gmacgrp_l3_l4_control3

This register controls the operations of the filter 0 of Layer 3 and Layer 4.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800490
i_emac_emac1	0xFF802000	0xFF802490
i_emac_emac2	0xFF804000	0xFF804490

Offset: 0x490

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved										l4dpim3 RW 0x0	l4dpm3 RW 0x0	l4spim3 RW 0x0	l4spm3 RW 0x0	Reserved	l4pen3 RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
l3hdbm3 RW 0x0					l3hsbm3 RW 0x0					l3daim3 RW 0x0	l3dam3 RW 0x0	l3saim3 RW 0x0	l3sam3 RW 0x0	Reserved	l3pen3 RW 0x0

gmacgrp_l3_l4_control3 Fields

Bit	Name	Description	Access	Reset
21	l4dpim3	When set, this bit indicates that the Layer 4 Destination Port number field is enabled for inverse matching. When reset, this bit indicates that the Layer 4 Destination Port number field is enabled for perfect matching. This bit is valid and applicable only when Bit 20 (L4DPM3) is set high.	RW	0x0

Bit	Name	Description	Access	Reset
20	l4dpm3	When set, this bit indicates that the Layer 4 Destination Port number field is enabled for matching. When reset, the MAC ignores the Layer 4 Destination Port number field for matching.	RW	0x0
19	l4spim3	When set, this bit indicates that the Layer 4 Source Port number field is enabled for inverse matching. When reset, this bit indicates that the Layer 4 Source Port number field is enabled for perfect matching. This bit is valid and applicable only when Bit 18 (L4SPM3) is set high.	RW	0x0
18	l4spm3	When set, this bit indicates that the Layer 4 Source Port number field is enabled for matching. When reset, the MAC ignores the Layer 4 Source Port number field for matching.	RW	0x0
16	l4pen3	When set, this bit indicates that the Source and Destination Port number fields for UDP frames are used for matching. When reset, this bit indicates that the Source and Destination Port number fields for TCP frames are used for matching. The Layer 4 matching is done only when either L4SPM3 or L4DPM3 bit is set high.	RW	0x0

Bit	Name	Description	Access	Reset
15:11	l3hdbm3	Layer 3 IP DA Higher Bits Match IPv4 Frames: This field contains the number of higher bits of IP Destination Address that are matched in the IPv4 frames. The following list describes the values of this field: * 0: No bits are masked. * 1: LSb[0] is masked. * 2: Two LSbs [1:0] are masked. * ... * 31: All bits except MSb are masked. IPv6 Frames: Bits [12:11] of this field correspond to Bits [6:5] of L3HSBM3, which indicate the number of lower bits of IP Source or Destination Address that are masked in the IPv6 frames. The following list describes the concatenated values of the L3HDBM3[1:0] and L3HSBM3 bits: * 0: No bits are masked. * 1: LSb[0] is masked. * 2: Two LSbs [1:0] are masked. * ... * 127: All bits except MSb are masked. This field is valid and applicable only if L3DAM3 or L3SAM3 is set high.	RW	0x0

Bit	Name	Description	Access	Reset
10:6	l3hsbm3	IPv4 Frames: This field contains the number of lower bits of IP Source Address that are masked for matching in the IPv4 frames. The following list describes the values of this field: * 0: No bits are masked. * 1: LSb[0] is masked. * 2: Two LSbs [1:0] are masked. * ... * 31: All bits except MSb are masked. IPv6 Frames: This field contains Bits [4:0] of the field that indicates the number of higher bits of IP Source or Destination Address matched in the IPv6 frames. This field is valid and applicable only if L3DAM3 or L3SAM3 is set high.	RW	0x0
5	l3daim3	When set, this bit indicates that the Layer 3 IP Destination Address field is enabled for inverse matching. When reset, this bit indicates that the Layer 3 IP Destination Address field is enabled for perfect matching. This bit is valid and applicable only when Bit 4 (L3DAM3) is set high.	RW	0x0
4	l3dam3	When set, this bit indicates that Layer 3 IP Destination Address field is enabled for matching. When reset, the MAC ignores the Layer 3 IP Destination Address field for matching. Note: When Bit 0 (L3PEN3) is set, you should set either this bit or Bit 2 (L3SAM3) because either IPv6 DA or SA can be checked for filtering.	RW	0x0

Bit	Name	Description	Access	Reset
3	l3saim3	When set, this bit indicates that the Layer 3 IP Source Address field is enabled for inverse matching. When reset, this bit indicates that the Layer 3 IP Source Address field is enabled for perfect matching. This bit is valid and applicable only when Bit 2 (L3SAM3) is set high.	RW	0x0
2	l3sam3	When set, this bit indicates that the Layer 3 IP Source Address field is enabled for matching. When reset, the MAC ignores the Layer 3 IP Source Address field for matching. Note: When Bit 0 (L3PEN3) is set, you should set either this bit or Bit 4 (L3DAM3) because either IPv6 SA or DA can be checked for filtering.	RW	0x0
0	l3pen3	When set, this bit indicates that the Layer 3 IP Source or Destination Address matching is enabled for the IPv6 frames. When reset, this bit indicates that the Layer 3 IP Source or Destination Address matching is enabled for the IPv4 frames. The Layer 3 matching is done only when either L3SAM3 or L3DAM3 bit is set high.	RW	0x0

gmacgrp_layer4_address3

Because the Layer 3 and Layer 4 Address Registers are double-synchronized to the Rx clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode of the Layer 3 and Layer 4 Address Registers are written. For proper synchronization updates, you should perform the consecutive writes to the same Layer 3 and Layer 4 Address Registers after at least four clock cycles delay of the destination clock.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800494

Module Instance	Base Address	Register Address
i_emac_emac1	0xFF802000	0xFF802494
i_emac_emac2	0xFF804000	0xFF804494

Offset: 0x494

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
l4dp3 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
l4sp3 RW 0x0															

gmacgrp_layer4_address3 Fields

Bit	Name	Description	Access	Reset
31:16	l4dp3	When Bit 16 (L4PEN3) is reset and Bit 20 (L4DPM3) is set in Register 292 (Layer 3 and Layer 4 Control Register 3), this field contains the value to be matched with the TCP Destination Port Number field in the IPv4 or IPv6 frames. When Bit 16 (L4PEN3) and Bit 20 (L4DPM3) are set in Register 292 (Layer 3 and Layer 4 Control Register 3), this field contains the value to be matched with the UDP Destination Port Number field in the IPv4 or IPv6 frames.	RW	0x0

Bit	Name	Description	Access	Reset
15:0	l4sp3	When Bit 16 (L4PEN3) is reset and Bit 20 (L4DPM3) is set in Register 292 (Layer 3 and Layer 4 Control Register 3), this field contains the value to be matched with the TCP Source Port Number field in the IPv4 or IPv6 frames. When Bit 16 (L4PEN3) and Bit 20 (L4DPM3) are set in Register 292 (Layer 3 and Layer 4 Control Register 3), this field contains the value to be matched with the UDP Source Port Number field in the IPv4 or IPv6 frames.	RW	0x0

gmacgrp_layer3_addr0_reg3

For IPv4 frames, the Layer 3 Address 0 Register 3 contains the 32-bit IP Source Address field. For IPv6 frames, it contains Bits [31:0] of the 128-bit IP Source Address or Destination Address field.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8004A0
i_emac_emac1	0xFF802000	0xFF8024A0
i_emac_emac2	0xFF804000	0xFF8044A0

Offset: 0x4A0

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
l3a03 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
l3a03 RW 0x0															

gmacgrp_layer3_addr0_reg3 Fields

Bit	Name	Description	Access	Reset
31:0	l3a03	When Bit 0 (L3PEN3) and Bit 2 (L3SAM3) are set in Register 292 (Layer 3 and Layer 4 Control Register 3), this field contains the value to be matched with Bits [31:0] of the IP Source Address field in the IPv6 frames. When Bit 0 (L3PEN3) and Bit 4 (L3DAM3) are set in Register 292 (Layer 3 and Layer 4 Control Register 3), this field contains the value to be matched with Bits [31:0] of the IP Destination Address field in the IPv6 frames. When Bit 0 (L3PEN3) is reset and Bit 2 (L3SAM3) is set in Register 292 (Layer 3 and Layer 4 Control Register 3), this field contains the value to be matched with the IP Source Address field in the IPv4 frames.	RW	0x0

gmacgrp_layer3_addr1_reg3

For IPv4 frames, the Layer 3 Address 1 Register 3 contains the 32-bit IP Destination Address field. For IPv6 frames, it contains Bits [63:32] of the 128-bit IP Source Address or Destination Address field.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8004A4
i_emac_emac1	0xFF802000	0xFF8024A4
i_emac_emac2	0xFF804000	0xFF8044A4

Offset: 0x4A4

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
l3a13 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
l3a13 RW 0x0															

gmacgrp_layer3_addr1_reg3 Fields

Bit	Name	Description	Access	Reset
31:0	l3a13	When Bit 0 (L3PEN3) and Bit 2 (L3SAM3) are set in Register 292 (Layer 3 and Layer 4 Control Register 3), this field contains the value to be matched with Bits [63:32] of the IP Source Address field in the IPv6 frames. When Bit 0 (L3PEN3) and Bit 4 (L3DAM3) are set in Register 292 (Layer 3 and Layer 4 Control Register 3), this field contains the value to be matched with Bits [63:32] of the IP Destination Address field in the IPv6 frames. When Bit 0 (L3PEN3) is reset and Bit 4 (L3DAM3) is set in Register 292 (Layer 3 and Layer 4 Control Register 3), this field contains the value to be matched with the IP Destination Address field in the IPv4 frames.	RW	0x0

gmacgrp_layer3_addr2_reg3

For IPv4 frames, the Layer 3 Address 2 Register 3 is reserved. For IPv6 frames, it contains Bits [95:64] of the 128-bit IP Source Address or Destination Address field.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8004A8
i_emac_emac1	0xFF802000	0xFF8024A8

Module Instance	Base Address	Register Address
i_emac_emac2	0xFF804000	0xFF8044A8

Offset: 0x4A8

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
l3a23 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
l3a23 RW 0x0															

gmacgrp_layer3_addr2_reg3 Fields

Bit	Name	Description	Access	Reset
31:0	l3a23	When Bit 0 (L3PEN3) and Bit 2 (L3SAM3) are set in Register 292 (Layer 3 and Layer 4 Control Register 3), this field contains the value to be matched with Bits [95:64] of the IP Source Address field in the IPv6 frames. When Bit 0 (L3PEN3) and Bit 4 (L3DAM3) are set in Register 292 (Layer 3 and Layer 4 Control Register 3), this field contains value to be matched with Bits [95:64] of the IP Destination Address field in the IPv6 frames. When Bit 0 (L3PEN3) is reset in Register 292 (Layer 3 and Layer 4 Control Register 3), this register is not used.	RW	0x0

gmacgrp_layer3_addr3_reg3

For IPv4 frames, the Layer 3 Address 3 Register 3 is reserved. For IPv6 frames, it contains Bits [127:96] of the 128-bit IP Source Address or Destination Address field.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8004AC
i_emac_emac1	0xFF802000	0xFF8024AC
i_emac_emac2	0xFF804000	0xFF8044AC

Offset: 0x4AC

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
l3a33 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
l3a33 RW 0x0															

gmacgrp_layer3_addr3_reg3 Fields

Bit	Name	Description	Access	Reset
31:0	l3a33	When Bit 0 (L3PEN3) and Bit 2 (L3SAM3) are set in Register 292 (Layer 3 and Layer 4 Control Register 3), this field contains the value to be matched with Bits [127:96] of the IP Source Address field in the IPv6 frames. When Bit 0 (L3PEN3) and Bit 4 (L3DAM3) are set in Register 292 (Layer 3 and Layer 4 Control Register 3), this field contains the value to be matched with Bits [127:96] of the IP Destination Address field in the IPv6 frames. When Bit 0 (L3PEN3) is reset in Register 292 (Layer 3 and Layer 4 Control Register 3), this register is not used.	RW	0x0

gmacgrp_hash_table_reg0

This register contains the first 32 bits of the hash table. The 256-bit Hash table is used for group address filtering. For hash

filtering, the content of the destination address in the incoming frame is passed through the CRC logic and the upper eight bits of the CRC register are used to index the content of the Hash table. The most significant bits determines the register to be used (Hash Table Register X), and the least significant five bits determine the bit within the register. For example, a hash value of 8b'10111111 selects Bit 31 of the Hash Table Register 5.

The hash value of the destination address is calculated in the following way:

1. Calculate the 32-bit CRC for the DA (See IEEE 802.3, Section 3.2.8 for the steps to calculate CRC32).
2. Perform bitwise reversal for the value obtained in Step 1.
3. Take the upper 8 bits from the value obtained in Step 2.

If the corresponding bit value of the register is 1'b1, the frame is accepted. Otherwise, it is rejected. If the Bit 1 (Pass All Multicast) is set in Register 1 (MAC Frame Filter), then all multicast frames are accepted regardless of the multicast hash values.
Because the Hash Table register is double-synchronized to the (G)MII clock domain, the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the Hash Table Register X registers are written.

Note: Because of double-synchronization, consecutive writes to this register should be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800500
i_emac_emac1	0xFF802000	0xFF802500
i_emac_emac2	0xFF804000	0xFF804500

Offset: 0x500

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ht31t0 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ht31t0 RW 0x0															

gmacgrp_hash_table_reg0 Fields

Bit	Name	Description	Access	Reset
31:0	ht31t0	This field contains the first 32 Bits (31:0) of the Hash table.	RW	0x0

gmacgrp_hash_table_reg1

This register contains the second 32 bits of the hash table.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800504
i_emac_emac1	0xFF802000	0xFF802504
i_emac_emac2	0xFF804000	0xFF804504

Offset: 0x504

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ht63t32 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ht63t32 RW 0x0															

gmacgrp_hash_table_reg1 Fields

Bit	Name	Description	Access	Reset
31:0	ht63t32	This field contains the second 32 Bits (63:32) of the Hash table.	RW	0x0

gmacgrp_hash_table_reg2

This register contains the third 32 bits of the hash table.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800508
i_emac_emac1	0xFF802000	0xFF802508
i_emac_emac2	0xFF804000	0xFF804508

Offset: 0x508

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ht95t64 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ht95t64 RW 0x0															

gmacgrp_hash_table_reg2 Fields

Bit	Name	Description	Access	Reset
31:0	ht95t64	This field contains the third 32 Bits (95:64) of the Hash table.	RW	0x0

gmacgrp_hash_table_reg3

This register contains the fourth 32 bits of the hash table.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF80050C
i_emac_emac1	0xFF802000	0xFF80250C
i_emac_emac2	0xFF804000	0xFF80450C

Offset: 0x50C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ht127t96 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ht127t96 RW 0x0															

gmacgrp_hash_table_reg3 Fields

Bit	Name	Description	Access	Reset
31:0	ht127t96	This field contains the fourth 32 Bits (127:96) of the Hash table.	RW	0x0

gmacgrp_hash_table_reg4

This register contains the fifth 32 bits of the hash table.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800510
i_emac_emac1	0xFF802000	0xFF802510
i_emac_emac2	0xFF804000	0xFF804510

Offset: 0x510

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ht159t128 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ht159t128 RW 0x0															

gmacgrp_hash_table_reg4 Fields

Bit	Name	Description	Access	Reset
31:0	ht159t128	This field contains the fifth 32 Bits (159:128) of the Hash table.	RW	0x0

gmacgrp_hash_table_reg5

This register contains the sixth 32 bits of the hash table.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800514
i_emac_emac1	0xFF802000	0xFF802514
i_emac_emac2	0xFF804000	0xFF804514

Offset: 0x514

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ht191t160 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ht191t160 RW 0x0															

gmacgrp_hash_table_reg5 Fields

Bit	Name	Description	Access	Reset
31:0	ht191t160	This field contains the sixth 32 Bits (191:160) of the Hash table.	RW	0x0

gmacgrp_hash_table_reg6

This register contains the seventh 32 bits of the hash table.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800518
i_emac_emac1	0xFF802000	0xFF802518
i_emac_emac2	0xFF804000	0xFF804518

Offset: 0x518

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ht223t196 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ht223t196 RW 0x0															

gmacgrp_hash_table_reg6 Fields

Bit	Name	Description	Access	Reset
31:0	ht223t196	This field contains the seventh 32 Bits (223:196) of the Hash table.	RW	0x0

gmacgrp_hash_table_reg7

This register contains the eighth 32 bits of the hash table.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF80051C
i_emac_emac1	0xFF802000	0xFF80251C
i_emac_emac2	0xFF804000	0xFF80451C

Offset: 0x51C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ht255t224 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ht255t224 RW 0x0															

gmacgrp_hash_table_reg7 Fields

Bit	Name	Description	Access	Reset
31:0	ht255t224	This field contains the eighth 32 Bits (255:224) of the Hash table.	RW	0x0

gmacgrp_vlan_incl_reg

Register 353 (VLAN Tag Inclusion or Replacement Register)

The VLAN Tag Inclusion or Replacement register contains the VLAN tag for insertion or replacement in the transmit frames.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800584
i_emac_emac1	0xFF802000	0xFF802584
i_emac_emac2	0xFF804000	0xFF804584

Offset: 0x584

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
reserved_31_20 RO 0x0												csvl RW 0x0	vlp RW 0x0	vlc RW 0x0	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
vlt RW 0x0															

gmacgrp_vlan_incl_reg Fields

Bit	Name	Description	Access	Reset
31:20	reserved_31_20	Reserved	RO	0x0
19	csvl	C-VLAN or S-VLAN When this bit is set, S-VLAN type (0x88A8) is inserted or replaced in the 13th and 14th bytes of transmitted frames. When this bit is reset, C-VLAN type (0x8100) is inserted or replaced in the transmitted frames.	RW	0x0
18	vlp	VLAN Priority Control When this bit is set, the control Bits [17:16] are used for VLAN deletion, insertion, or replacement. When this bit is reset, the mti_vlan_ctrl_i control input is used, and Bits [17:16] are ignored.	RW	0x0

Bit	Name	Description	Access	Reset
17:16	vlc	VLAN Tag Control in Transmit Frames * 2'b00: No VLAN tag deletion, insertion, or replacement * 2'b01: VLAN tag deletion The MAC removes the VLAN type (bytes 13 and 14) and VLAN tag (bytes 15 and 16) of all transmitted frames with VLAN tags. * 2'b10: VLAN tag insertion The MAC inserts VLT in bytes 15 and 16 of the frame after inserting the Type value (0x8100/0x88a8) in bytes 13 and 14. This operation is performed on all transmitted frames, irrespective of whether they already have a VLAN tag. * 2'b11: VLAN tag replacement The MAC replaces VLT in bytes 15 and 16 of all VLAN-type transmitted frames (Bytes 13 and 14 are 0x8100/0x88a8). Note: Changes to this field take effect only on the start of a frame. If you write this register field when a frame is being transmitted, only the subsequent frame can use the updated value, that is, the current frame does not use the updated value.	RW	0x0
15:0	vlt	VLAN Tag for Transmit Frames This field contains the value of the VLAN tag to be inserted or replaced. The value must only be changed when the transmit lines are inactive or during the initialization phase. Bits[15:13] are the User Priority, Bit 12 is the CFI/DEI, and Bits[11:0] are the VLAN tag's VID field.	RW	0x0

gmacgrp_vlan_hash_table_reg

The 16-bit Hash table is used for group address filtering based on VLAN tag when Bit 18 (VTHM) of Register 7 (VLAN Tag Register) is set. For hash filtering, the content of the 16-bit VLAN tag or 12-bit VLAN ID (based on Bit 16 (ETV) of VLAN Tag Register) in the incoming frame

is passed through the CRC logic and the upper four bits of the calculated CRC are used to index the contents of the VLAN Hash table. For example, a hash value of 4b'1000 selects Bit 8 of the VLAN Hash table.
The hash value of the destination address is calculated in the following way:

1. Calculate the 32-bit CRC for the VLAN tag or ID (See IEEE 802.3, Section 3.2.8 for the steps to calculate CRC32).
2. Perform bitwise reversal for the value obtained in Step 1.
3. Take the upper four bits from the value obtained in Step 2.

If the corresponding bit value of the register is 1'b1, the frame is accepted. Otherwise, it is rejected. Because the Hash Table register is double-synchronized to the (G)MII clock domain, the synchronization is triggered only when Bits[15:8] (in little-endian mode) of this register are written.

Notes:
 * Because of double-synchronization, consecutive writes to this register should be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800588
i_emac_emac1	0xFF802000	0xFF802588
i_emac_emac2	0xFF804000	0xFF804588

Offset: 0x588

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
vlht RW 0x0															

gmacgrp_vlan_hash_table_reg Fields

Bit	Name	Description	Access	Reset
15:0	vlht	This field contains the 16-bit VLAN Hash Table.	RW	0x0

gmacgrp_timestamp_control

Register 448 (Timestamp Control Register)

This register controls the operation of the System Time generator and the processing of PTP packets for timestamping in the Receiver.

Note:
 * Bits[5:1] are reserved when External Timestamp Input feature is enabled.
 * Bits[19:8] are reserved and read-only when Advanced Timestamp feature is not enabled.
 * Bits[28:24] are reserved and read-only when Auxiliary Snapshot feature is not enabled.
 * Release 3.60a onwards, the functions of Bits 17 and 16 (SNAPTYPSEL) have changed. These functions are not backward compatible with the functions described in release 3.50a.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800700
i_emac_emac1	0xFF802000	0xFF802700
i_emac_emac2	0xFF804000	0xFF804700

Offset: 0x700

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
reserved_31_29 RO 0x0			atsen3 RO 0x0	atsen2 RO 0x0	atsen1 RO 0x0	atsen0 RO 0x0	atsfc RO 0x0	reserved_23_19 RO 0x0					tsenmacaddr RW 0x0	snaptypsel RW 0x0	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
tsmstrena RW 0x0	tsevntena RW 0x0	tsipv4ena RW 0x1	tsipv6ena RW 0x0	tsipena RW 0x0	tsver2ena RW 0x0	tsctrlssr RW 0x0	tsenall RW 0x0	reserved_7_6 RO 0x0		tsaddreg RO 0x0	tstrig RO 0x0	tsupdt RO 0x0	tsinit RO 0x0	tscfupdt RO 0x0	tsena RW 0x0

gmacgrp_timestamp_control Fields

Bit	Name	Description	Access	Reset
31:29	reserved_31_29	Reserved	RO	0x0
28	atsen3	Auxiliary Snapshot 3 Enable This field controls capturing the Auxiliary Snapshot Trigger 3. When this bit is set, the Auxiliary snapshot of event on ptp_aux_trig_i[3] input is enabled. When this bit is reset, the events on this input are ignored.	RO	0x0
27	atsen2	Auxiliary Snapshot 2 Enable This field controls capturing the Auxiliary Snapshot Trigger 2. When this bit is set, the Auxiliary snapshot of event on ptp_aux_trig_i[2] input is enabled. When this bit is reset, the events on this input are ignored.	RO	0x0

Bit	Name	Description	Access	Reset
26	atsen1	Auxiliary Snapshot 1 Enable This field controls capturing the Auxiliary Snapshot Trigger 1. When this bit is set, the Auxiliary snapshot of event on ptp_aux_trig_i[1] input is enabled. When this bit is reset, the events on this input are ignored.	RO	0x0
25	atsen0	Auxiliary Snapshot 0 Enable This field controls capturing the Auxiliary Snapshot Trigger 0. When this bit is set, the Auxiliary snapshot of event on ptp_aux_trig_i[0] input is enabled. When this bit is reset, the events on this input are ignored. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RO	0x0
24	atsfc	Auxiliary Snapshot FIFO Clear When set, it resets the pointers of the Auxiliary Snapshot FIFO. This bit is cleared when the pointers are reset and the FIFO is empty. When this bit is high, auxiliary snapshots get stored in the FIFO. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RO	0x0
23:19	reserved_23_19	Reserved	RO	0x0

Bit	Name	Description	Access	Reset
18	tsenmacaddr	Enable MAC address for PTP Frame Filtering When set, the DA MAC address (that matches any MAC Address register) is used to filter the PTP frames when PTP is directly sent over Ethernet. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
17:16	snaptypsel	Select PTP packets for Taking Snapshots These bits along with Bits 15 and 14 decide the set of PTP packet types for which snapshot needs to be taken.	RW	0x0
15	tsmstrena	Enable Snapshot for Messages Relevant to Master When set, the snapshot is taken only for the messages relevant to the master node. Otherwise, the snapshot is taken for the messages relevant to the slave node. **Value** **Description** 0x0 SLAVE 0x1 MASTER	RW	0x0

Bit	Name	Description	Access	Reset
14	tsevntena	Enable Timestamp Snapshot for Event Messages When set, the timestamp snapshot is taken only for event messages (SYNC, Delay_Req, Pdelay_Req, or Pdelay_Resp). When reset, the snapshot is taken for all messages except Announce, Management, and Signaling. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
13	tsipv4ena	Enable Processing of PTP Frames Sent over IPv4-UDP When set, the MAC receiver processes the PTP packets encapsulated in UDP over IPv4 packets. When this bit is clear, the MAC ignores the PTP transported over UDP-IPv4 packets. This bit is set by default. **Value** **Description** 0x0 NO_PROCESS_PTP 0x1 PROCESS_PTP	RW	0x1
12	tsipv6ena	Enable Processing of PTP Frames Sent Over IPv6-UDP When set, the MAC receiver processes PTP packets encapsulated in UDP over IPv6 packets. When this bit is clear, the MAC ignores the PTP transported over UDP-IPv6 packets. **Value** **Description** 0x0 NO_PROCESS_PTP 0x1 PROCESS_PTP	RW	0x0

Bit	Name	Description	Access	Reset
11	tsipena	Enable Processing of PTP over Ethernet Frames When set, the MAC receiver processes the PTP packets encapsulated directly in the Ethernet frames. When this bit is clear, the MAC ignores the PTP over Ethernet packets. **Value** **Description** 0x0 NO_PROCESS_PTP 0x1 PROCESS_PTP	RW	0x0
10	tsver2ena	Enable PTP packet Processing for Version 2 Format When set, the PTP packets are processed using the 1588 version 2 format. Otherwise, the PTP packets are processed using the version 1 format. **Value** **Description** 0x0 PTP_1588_VER1 0x1 PTP_1588_VER2	RW	0x0
9	tsctrlssr	Timestamp Digital or Binary Rollover Control When set, the Timestamp Low register rolls over after 0x3B9A_C9FF value (that is, 1 nanosecond accuracy) and increments the timestamp (High) seconds. When reset, the rollover value of sub-second register is 0x7FFF_FFFF. The sub-second increment has to be programmed correctly depending on the PTP reference clock frequency and the value of this bit. **Value** **Description** 0x0 NOTIMESTAMP_LOW_ROLL_MAX 0x1 TIMESTAMP_LOW_ROLL_1NS	RW	0x0

Bit	Name	Description	Access	Reset
8	tsenall	Enable Timestamp for All Frames When set, the timestamp snapshot is enabled for all frames received by the MAC. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
7:6	reserved_7_6	Reserved	RO	0x0
5	tsaddreg	Addend Reg Update When set, the content of the Timestamp Addend register is updated in the PTP block for fine correction. This is cleared when the update is completed. This register bit should be zero before setting it. **Value** **Description** 0x0 NOTIMESTAMP_ADDEND_UPDATED 0x1 TIMESTAMP_ADDEND_UPDATED	RO	0x0
4	tstrig	Timestamp Interrupt Trigger Enable When set, the timestamp interrupt is generated when the System Time becomes greater than the value written in the Target Time register. This bit is reset after the generation of the Timestamp Trigger Interrupt. **Value** **Description** 0x0 NOTIMESTAMP_INTR_TRIG_EN 0x1 TIMESTAMP_INTR_TRIG_EN	RO	0x0

Bit	Name	Description	Access	Reset
3	tsupdt	Timestamp Update When set, the system time is updated (added or subtracted) with the value specified in Register 452 (System Time - Seconds Update Register) and Register 453 (System Time - Nanoseconds Update Register). This bit should be read zero before updating it. This bit is reset when the update is completed in hardware. The Timestamp Higher Word register is not updated. **Value** — **Description** 0x0 — NOTIMESTAMP_UPDATED 0x1 — TIMESTAMP_UPDATED	RO	0x0
2	tsinit	Timestamp Initialize When set, the system time is initialized (overwritten) with the value specified in the Register 452 (System Time - Seconds Update Register) and Register 453 (System Time - Nanoseconds Update Register). This bit should be read zero before updating it. This bit is reset when the initialization is complete. The Timestamp Higher Word register can only be initialized. **Value** — **Description** 0x0 — NOTIMESTAMP_INIT 0x1 — TIMESTAMP_INIT	RO	0x0

Bit	Name	Description	Access	Reset
1	tscfupdt	Timestamp Fine or Coarse Update When set, this bit indicates that the system times update should be done using the fine update method. When reset, it indicates the system timestamp update should be done using the Coarse method. **Value** **Description** 0x0 TIMESTAMP_COARSE 0x1 TIMESTAMP_FINE	RO	0x0
0	tsena	Timestamp Enable When set, the timestamp is added for the transmit and receive frames. When disabled, timestamp is not added for the transmit and receive frames and the Timestamp Generator is also suspended. You need to initialize the Timestamp (system time) after enabling this mode. On the receive side, the MAC processes the 1588 frames only if this bit is set. **Value** **Description** 0x0 NOTIMESTAMP 0x1 TIMESTAMP	RW	0x0

gmacgrp_sub_second_increment

In the Coarse Update mode (TSCFUPDT bit in Register 448), the value in this register is added to the system time every clock cycle of clk_ptp_ref_i. In the Fine Update mode, the value in this register is added to the system time whenever the Accumulator gets an overflow.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800704
i_emac_emac1	0xFF802000	0xFF802704

Module Instance	Base Address	Register Address
i_emac_emac2	0xFF804000	0xFF804704

Offset: 0x704

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								ssinc RW 0x0							

gmacgrp_sub_second_increment Fields

Bit	Name	Description	Access	Reset
7:0	ssinc	The value programmed in this field is accumulated every clock cycle (of clk_ptp_i) with the contents of the sub-second register. For example, when PTP clock is 50 MHz (period is 20 ns), you should program 20 (0x14) when the System Time-Nanoseconds register has an accuracy of 1 ns (TSCTRLSSR bit is set). When TSCTRLSSR is clear, the Nanoseconds register has a resolution of ~0.465ns. In this case, you should program a value of 43 (0x2B) that is derived by 20ns/0.465.	RW	0x0

gmacgrp_system_time_seconds

The System Time -Seconds register, along with System-TimeNanoseconds register, indicates the current value of the system time maintained by the MAC. Though it is updated on a continuous basis, there is some delay from the actual time because of clock domain transfer latencies (from clk_ptp_ref_i to l3_sp_clk).

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800708
i_emac_emac1	0xFF802000	0xFF802708
i_emac_emac2	0xFF804000	0xFF804708

Offset: 0x708

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
tss RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
tss RO 0x0															

gmacgrp_system_time_seconds Fields

Bit	Name	Description	Access	Reset
31:0	tss	The value in this field indicates the current value in seconds of the System Time maintained by the MAC.	RO	0x0

gmacgrp_system_time_nanoseconds

The value in this field has the sub second representation of time, with an accuracy of 0.46 ns. When TSCTRLSSR is set, each bit represents 1 ns and the maximum value is 0x3B9A_C9FF, after which it rolls-over to zero.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF80070C
i_emac_emac1	0xFF802000	0xFF80270C
i_emac_emac2	0xFF804000	0xFF80470C

Offset: 0x70C

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	tsss RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
tsss RO 0x0															

gmacgrp_system_time_nanoseconds Fields

Bit	Name	Description	Access	Reset
30:0	tsss	The value in this field has the sub second representation of time, with an accuracy of 0.46 ns. When bit 9 (TSCTRLSSR) is set in Register 448 (Timestamp Control Register), each bit represents 1 ns and the maximum value is 0x3B9A_C9FF, after which it rolls-over to zero.	RO	0x0

gmacgrp_system_time_seconds_update

The System Time - Seconds Update register, along with the System Time - Nanoseconds Update register, initializes or updates the system time maintained by the MAC. You must write both of these registers before setting the TSINIT or TSUPDT bits in the Timestamp Control register.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800710
i_emac_emac1	0xFF802000	0xFF802710
i_emac_emac2	0xFF804000	0xFF804710

Offset: 0x710

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
tss RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
tss RW 0x0															

gmacgrp_system_time_seconds_update Fields

Bit	Name	Description	Access	Reset
31:0	tss	The value in this field indicates the time in seconds to be initialized or added to the system time.	RW	0x0

gmacgrp_system_time_nanoseconds_update

Update system time

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800714
i_emac_emac1	0xFF802000	0xFF802714
i_emac_emac2	0xFF804000	0xFF804714

Offset: 0x714

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addsub RW 0x0	tsss RW 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
tsss RW 0x0															

gmacgrp_system_time_nanoseconds_update Fields

Bit	Name	Description	Access	Reset
31	addsub	When this bit is set, the time value is subtracted with the contents of the update register. When this bit is reset, the time value is added with the contents of the update register. Value: Description 0x0: DISABLED 0x1: ENABLED	RW	0x0
30:0	tsss	The value in this field has the sub second representation of time, with an accuracy of 0.46 ns. When bit 9 (TSCTRLSSR) is set in Register 448 (Timestamp Control Register), each bit represents 1 ns and the programmed value should not exceed 0x3B9A_C9FF.	RW	0x0

gmacgrp_timestamp_addend

This register value is used only when the system time is configured for Fine Update mode (TSCFUPDT bit in Register 448). This register content is added to a 32-bit accumulator in every clock cycle (of clk_ptp_ref_i) and the system time is updated whenever the accumulator overflows.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800718
i_emac_emac1	0xFF802000	0xFF802718
i_emac_emac2	0xFF804000	0xFF804718

Offset: 0x718

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
tsar RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
tsar RW 0x0															

gmacgrp_timestamp_addend Fields

Bit	Name	Description	Access	Reset
31:0	tsar	This field indicates the 32-bit time value to be added to the Accumulator register to achieve time synchronization.	RW	0x0

gmacgrp_target_time_seconds

The Target Time Seconds register, along with Target Time Nanoseconds register, is used to schedule an interrupt event (Register 458[1] when Advanced Timestamping is enabled; otherwise, TS interrupt bit in Register14[9]) when the system time exceeds the value programmed in these registers.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF80071C
i_emac_emac1	0xFF802000	0xFF80271C
i_emac_emac2	0xFF804000	0xFF80471C

Offset: 0x71C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
tstr RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
tstr RW 0x0															

gmacgrp_target_time_seconds Fields

Bit	Name	Description	Access	Reset
31:0	tstr	This register stores the time in seconds. When the timestamp value matches or exceeds both Target Timestamp registers, then based on Bits [6:5] of Register 459 (PPS Control Register), the MAC starts or stops the PPS signal output and generates an interrupt (if enabled).	RW	0x0

gmacgrp_target_time_nanoseconds

Target time

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800720
i_emac_emac1	0xFF802000	0xFF802720
i_emac_emac2	0xFF804000	0xFF804720

Offset: 0x720

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
trgtbusy RO 0x0	ttslo RW 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ttslo RW 0x0															

gmacgrp_target_time_nanoseconds Fields

Bit	Name	Description	Access	Reset
31	trgtbusy	The MAC sets this bit when the PPSCMD field (Bits[3:0]) in Register 459 (PPS Control Register) is programmed to 010 or 011. Programming the PPSCMD field to 010 or 011, instructs the MAC to synchronize the Target Time Registers to the PTP clock domain. The MAC clears this bit after synchronizing the Target Time Registers to the PTP clock domain The application must not update the Target Time Registers when this bit is read as 1. Otherwise, the synchronization of the previous programmed time gets corrupted. This bit is reserved when the Enable Flexible Pulse-Per-Second Output feature is not selected.	RO	0x0

Bit	Name	Description	Access	Reset
30:0	ttslo	This register stores the time in (signed) nanoseconds. When the value of the timestamp matches the both Target Timestamp registers, then based on the TRGTMODSEL0 field (Bits [6:5]) in Register 459 (PPS Control Register), the MAC starts or stops the PPS signal output and generates an interrupt (if enabled). This value should not exceed 0x3B9A_C9FF when TSCTRLSSR is set in the Timestamp control register. The actual start or stop time of the PPS signal output may have an error margin up to one unit of sub-second increment value.	RW	0x0

gmacgrp_system_time_higher_word_seconds

System time higher word

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800724
i_emac_emac1	0xFF802000	0xFF802724
i_emac_emac2	0xFF804000	0xFF804724

Offset: 0x724

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
tshwr RW 0x0															

gmacgrp_system_time_higher_word_seconds Fields

Bit	Name	Description	Access	Reset
15:0	tshwr	This field contains the most significant 16-bits of the timestamp seconds value. The register is directly written to initialize the value. This register is incremented when there is an overflow from the 32-bits of the System Time - Seconds register.	RW	0x0

gmacgrp_timestamp_status

Timestamp status. All bits except Bits[27:25] get cleared when the host reads this register.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800728
i_emac_emac1	0xFF802000	0xFF802728
i_emac_emac2	0xFF804000	0xFF804728

Offset: 0x728

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		atsns RO 0x0					atsstm RO 0x0	Reserved				atsstn RO 0x0			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												tstrgterr RO 0x0	auxtstrig RO 0x0	tstargt RO 0x0	tssovf RO 0x0

gmacgrp_timestamp_status Fields

Bit	Name	Description	Access	Reset
29:25	atsns	This field indicates the number of Snapshots available in the FIFO. A value of 16 (equal to the depth of the FIFO) indicates that the Auxiliary Snapshot FIFO is full. These bits are cleared (to 00000) when the Auxiliary snapshot FIFO clear bit is set.	RO	0x0
24	atsstm	This bit is set when the Auxiliary timestamp snapshot FIFO is full and external trigger was set. This indicates that the latest snapshot is not stored in the FIFO. **Value** — **Description** 0x0 — NOTFULL 0x1 — FULL	RO	0x0
19:16	atsstn	These bits identify the Auxiliary trigger inputs for which the timestamp available in the Auxiliary Snapshot Register is applicable. When more than one bit is set at the same time, it means that corresponding auxiliary triggers were sampled at the same clock. These bits are applicable only if the number of Auxiliary snapshots is more than one. One bit is assigned for each trigger as shown in the following list: * Bit 16: Auxiliary trigger 0 * Bit 17: Auxiliary trigger 1 * Bit 18: Auxiliary trigger 2 * Bit 19: Auxiliary trigger 3 The software can read this register to find the triggers that are set when the timestamp is taken.	RO	0x0

Bit	Name	Description	Access	Reset
3	tstrgterr	This bit is set when the target time, being programmed in Target Time Registers, is already elapsed. This bit is cleared when read by the application. **Value** **Description** 0x0 RESET 0x1 SET	RO	0x0
2	auxtstrig	This bit is set high when the auxiliary snapshot is written to the FIFO. **Value** **Description** 0x0 RESET 0x1 SET	RO	0x0
1	tstargt	When set, this bit indicates that the value of system time is greater or equal to the value specified in the Register 455 (Target Time Seconds Register) and Register 456 (Target Time Nanoseconds Register). **Value** **Description** 0x0 RESET 0x1 SET	RO	0x0
0	tssovf	When set, this bit indicates that the seconds value of the timestamp (when supporting version 2 format) has overflowed beyond 32'hFFFF_FFFF. **Value** **Description** 0x0 RESET 0x1 SET	RO	0x0

gmacgrp_pps_control

Controls timestamp Pulse-Per-Second output

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF80072C
i_emac_emac1	0xFF802000	0xFF80272C
i_emac_emac2	0xFF804000	0xFF80472C

Offset: 0x72C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									trgtmodsel0 RW 0x0		ppsen 0 RW 0x0	ppsctrl_ppscmd RW 0x0			

gmacgrp_pps_control Fields

Bit	Name	Description	Access	Reset
6:5	trgtmodsel0	This field indicates the Target Time registers (register 455 and 456) mode for PPS0 output signal Value: Description 0x0: TRGTINTERONLY 0x2: TRGTINTPPS0 0x3: TRGTNOINTER	RW	0x0
4	ppsen0	When set low, Bits[3:0] function as PPSCTRL (backward compatible). When set high, Bits[3:0] function as PPSCMD. Value: Description 0x0: PPSCTRL 0x1: PPSCMD	RW	0x0

Bit	Name	Description	Access	Reset

3:0	ppsctrl_ppscmd	PPSCTRL0: PPS0 Output Frequency Control
		This field controls the frequency of the PPS0 output (ptp_pps_o[0]) signal. The default value of PPSCTRL is 0000, and the PPS output is 1 pulse (of width clk_ptp_i) every second. For other values of PPSCTRL, the PPS output becomes a generated clock of following frequencies:
		• 0001: The binary rollover is 2 Hz, and the digital rollover is 1 Hz.
		• 0010: The binary rollover is 4 Hz, and the digital rollover is 2 Hz.
		• 0011: The binary rollover is 8 Hz, and the digital rollover is 4 Hz.
		• 0100: The binary rollover is 16 Hz, and the digital rollover is 8 Hz.
		• ...
		• 1111: The binary rollover is 32.768 KHz, and the digital rollover is 16.384 KHz.
		Note: In the binary rollover mode, the PPS output (ptp_pps_o) has a duty cycle of 50 percent with these frequencies.
		In the digital rollover mode, the PPS output frequency is an average number. The actual clock is of different frequency that gets synchronized every second. For example:
		• When PPSCTRL = 0001, the PPS (1 Hz) has a low period of 537 ms and a high period of 463 ms
		• When PPSCTRL = 0010, the PPS (2 Hz) is a sequence of:
		  • One clock of 50 percent duty cycle and 537 ms period>
		  • Second clock of 463 ms period (268 ms low and 195 ms high)
		• When PPSCTRL = 0011, the PPS (4 Hz) is a sequence of:
		  • Three clocks of 50 percent duty cycle and 268 ms period
		  • Fourth clock of 195 ms period (134 ms low and 61 ms high)
		This behavior is because of the non-linear toggling of bits in the digital rollover mode in Register 451 (System Time - Nanoseconds Register).
		Flexible PPS0 Output (ptp_pps_o[0]) Control
			RW	0x0

Bit	Name	Description	Access	Reset
		Programming these bits with a non-zero value instructs the MAC to initiate an event. Once the command is transferred or synchronized to the PTP clock domain, these bits get cleared automatically. The Software should ensure that these bits are programmed only when they are all-zero. The following list describes the values of PPSCMD0: • 0000: No Command • 0001: START Single Pulse This command generates single pulse rising at the start point defined in Target Time Registers (register 455 and 456) and of a duration defined in the PPS0 Width Register. • 0010: START Pulse Train This command generates the train of pulses rising at the start point defined in the Target Time Registers and of a duration defined in the PPS0 Width Register and repeated at interval defined in the PPS Interval Register. By default, the PPS pulse train is free-running unless stopped by 'STOP Pulse train at time' or 'STOP Pulse Train immediately' commands. • 0011: Cancel START This command cancels the START Single Pulse and START Pulse Train commands if the system time has not crossed the programmed start time. • 0100: STOP Pulse train at time This command stops the train of pulses initiated by the START Pulse Train command (PPSCMD = 0010) after the time programmed in the Target Time registers elapses. • 0101: STOP Pulse Train immediately This command immediately stops the train of pulses initiated by the START Pulse Train command (PPSCMD = 0010). • 0110: Cancel STOP Pulse train This command cancels the STOP pulse train at time command if the programmed stop time has not elapsed. The PPS pulse train becomes free-running on the successful execution of this command. • 0111-1111: Reserved		

gmacgrp_auxiliary_timestamp_nanoseconds

This register, along with Register 461 (Auxiliary Timestamp Seconds Register), gives the 64-bit timestamp stored as auxiliary snapshot. The two registers together form the read port of a 64-bit wide FIFO with a depth of 16. Multiple snapshots can be stored in this FIFO. The ATSNS bits in the Timestamp Status register indicate the fill-level of this FIFO. The top of the FIFO is removed only when the last byte of Register 461 (Auxiliary Timestamp - Seconds Register) is read. In the little-endian mode, this means when Bits[31:24] are read.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800730
i_emac_emac1	0xFF802000	0xFF802730
i_emac_emac2	0xFF804000	0xFF804730

Offset: 0x730

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	auxtslo RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
auxtslo RO 0x0															

gmacgrp_auxiliary_timestamp_nanoseconds Fields

Bit	Name	Description	Access	Reset
30:0	auxtslo	Contains the lower 32 bits (nano-seconds field) of the auxiliary timestamp.	RO	0x0

gmacgrp_auxiliary_timestamp_seconds

Contains the higher 32 bits (Seconds field) of the auxiliary

timestamp.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800734
i_emac_emac1	0xFF802000	0xFF802734
i_emac_emac2	0xFF804000	0xFF804734

Offset: 0x734

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
auxtshi RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
auxtshi RO 0x0															

gmacgrp_auxiliary_timestamp_seconds Fields

Bit	Name	Description	Access	Reset
31:0	auxtshi	Contains the higher 32 bits (Seconds field) of the auxiliary timestamp.	RO	0x0

gmacgrp_pps0_interval

The PPS0 Interval register contains the number of units of sub-second increment value between the rising edges of PPS0 signal output (ptp_pps_o[0]).

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800760
i_emac_emac1	0xFF802000	0xFF802760
i_emac_emac2	0xFF804000	0xFF804760

Offset: 0x760

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ppsint RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ppsint RW 0x0															

gmacgrp_pps0_interval Fields

Bit	Name	Description	Access	Reset
31:0	ppsint	These bits store the interval between the rising edges of PPS0 signal output in terms of units of sub-second increment value. You need to program one value less than the required interval. For example, if the PTP reference clock is 50 MHz (period of 20ns), and desired interval between rising edges of PPS0 signal output is 100ns (that is, five units of sub-second increment value), then you should program value 4 (5 -1) in this register.	RW	0x0

gmacgrp_pps0_width

The PPS0 Width register contains the number of units of sub-second increment value between the rising and corresponding falling edges of the PPS0 signal output (ptp_pps_o[0]).

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800764
i_emac_emac1	0xFF802000	0xFF802764
i_emac_emac2	0xFF804000	0xFF804764

Offset: 0x764

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ppswidth RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ppswidth RW 0x0															

gmacgrp_pps0_width Fields

Bit	Name	Description	Access	Reset
31:0	ppswidth	These bits store the width between the rising edge and corresponding falling edge of the PPS0 signal output in terms of units of sub-second increment value. You need to program one value less than the required interval. For example, if PTP reference clock is 50 MHz (period of 20ns), and desired width between the rising and corresponding falling edges of PPS0 signal output is 80ns (that is, four units of sub-second increment value), then you should program value 3 (4-1) in this register. Note: The value programmed in this register must be lesser than the value programmed in Register 472 (PPS0 Interval Register).	RW	0x0

gmacgrp_mac_address16_high

Register 512 (MAC Address16 High Register)

The MAC Address16 High register holds the upper 16 bits of the 17th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address16 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address16 Low Register must be performed after at least four clock

cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800800
i_emac_emac1	0xFF802000	0xFF802800
i_emac_emac2	0xFF804000	0xFF804800

Offset: 0x800

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address16_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 17th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
30	sa	Source Address When this bit is set, the MAC Address16[47:0] is used to compare with the SA fields of the received frame. When this bit is reset, the MAC Address16[47:0] is used to compare with the DA fields of the received frame. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
29	mbc_5	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address16 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0

Bit	Name	Description	Access	Reset
28	mbc_4	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address16 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0
27	mbc_3	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address16 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0

Bit	Name	Description	Access	Reset
26	mbc_2	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address16 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0
25	mbc_1	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address16 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0

Bit	Name	Description	Access	Reset
24	mbc_0	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address16 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). Value: Description 0x0: UNMASKED 0x1: MASKED	RW	0x0
23:16	reserved_23_16	Reserved	RO	0x0
15:0	addrhi	MAC Address16 [47:32] This field contains the upper 16 bits (47:32) of the 17th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address16_low

Register 513 (MAC Address16 Low Register)

The MAC Address16 Low register holds the lower 32 bits of the 17th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800804
i_emac_emac1	0xFF802000	0xFF802804
i_emac_emac2	0xFF804000	0xFF804804

Offset: 0x804

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address16_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address16 [31:0] This field contains the lower 32 bits of the 17th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address17_high

Register 514 (MAC Address17 High Register)

The MAC Address17 High register holds the upper 16 bits of the 18th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address17 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address17 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800808
i_emac_emac1	0xFF802000	0xFF802808
i_emac_emac2	0xFF804000	0xFF804808

Offset: 0x808

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address17_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 18th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
30	sa	Source Address When this bit is set, the MAC Address17[47:0] is used to compare with the SA fields of the received frame. When this bit is reset, the MAC Address17[47:0] is used to compare with the DA fields of the received frame. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
29	mbc_5	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address17 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0
28	mbc_4	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address17 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0

Bit	Name	Description	Access	Reset
27	mbc_3	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address17 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0
26	mbc_2	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address17 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0

Bit	Name	Description	Access	Reset
25	mbc_1	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address17 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0
24	mbc_0	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address17 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0
23:16	reserved_23_16	Reserved	RO	0x0
15:0	addrhi	MAC Address18 [47:32] This field contains the upper 16 bits (47:32) of the 19th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address17_low

Register 515 (MAC Address17 Low Register)

The MAC Address17 Low register holds the lower 32 bits of the 18th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF80080C
i_emac_emac1	0xFF802000	0xFF80280C
i_emac_emac2	0xFF804000	0xFF80480C

Offset: 0x80C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address17_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address17 [31:0] This field contains the lower 32 bits of the 18th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address18_high

Register 516 (MAC Address18 High Register)

The MAC Address18 High register holds the upper 16 bits of the 19th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address18 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address18 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800810
i_emac_emac1	0xFF802000	0xFF802810
i_emac_emac2	0xFF804000	0xFF804810

Offset: 0x810

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address18_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 19th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30	sa	Source Address When this bit is set, the MAC Address18[47:0] is used to compare with the SA fields of the received frame. When this bit is reset, the MAC Address18[47:0] is used to compare with the DA fields of the received frame. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
29	mbc_5	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address18 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0
28	mbc_4	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address18 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0

Bit	Name	Description	Access	Reset
27	mbc_3	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address18 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0
26	mbc_2	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address18 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0

Bit	Name	Description	Access	Reset
25	mbc_1	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address18 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0
24	mbc_0	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address18 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0
23:16	reserved_23_16	Reserved	RO	0x0
15:0	addrhi	MAC Address18 [47:32] This field contains the upper 16 bits (47:32) of the 19th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address18_low

Register 517 (MAC Address18 Low Register)

The MAC Address18 Low register holds the lower 32 bits of the 19th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800814
i_emac_emac1	0xFF802000	0xFF802814
i_emac_emac2	0xFF804000	0xFF804814

Offset: 0x814

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address18_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address18 [31:0] This field contains the lower 32 bits of the 19th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address19_high

Register 518 (MAC Address19 High Register)

The MAC Address19 High register holds the upper 16 bits of the 20th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address19 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address19 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800818
i_emac_emac1	0xFF802000	0xFF802818
i_emac_emac2	0xFF804000	0xFF804818

Offset: 0x818

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address19_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 20th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30	sa	Source Address When this bit is set, the MAC Address19[47:0] is used to compare with the SA fields of the received frame. When this bit is reset, the MAC Address19[47:0] is used to compare with the DA fields of the received frame. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
29	mbc_5	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address19 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0
28	mbc_4	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address19 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0

Bit	Name	Description	Access	Reset
27	mbc_3	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address19 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0
26	mbc_2	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address19 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0

Bit	Name	Description	Access	Reset
25	mbc_1	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address19 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0
24	mbc_0	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address19 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0
23:16	reserved_23_16	Reserved	RO	0x0
15:0	addrhi	MAC Address19 [47:32] This field contains the upper 16 bits (47:32) of the 20th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address19_low

Register 519 (MAC Address19 Low Register)

The MAC Address19 Low register holds the lower 32 bits of the 20th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF80081C
i_emac_emac1	0xFF802000	0xFF80281C
i_emac_emac2	0xFF804000	0xFF80481C

Offset: 0x81C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address19_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address19 [31:0] This field contains the lower 32 bits of the 20th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address20_high

Register 520 (MAC Address20 High Register)

The MAC Address20 High register holds the upper 16 bits of the 21st 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address20 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address20 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800820
i_emac_emac1	0xFF802000	0xFF802820
i_emac_emac2	0xFF804000	0xFF804820

Offset: 0x820

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address20_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 21st MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30	sa	Source Address When this bit is set, the MAC Address20[47:0] is used to compare with the SA fields of the received frame. When this bit is reset, the MAC Address20[47:0] is used to compare with the DA fields of the received frame. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
29	mbc_5	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address20 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0
28	mbc_4	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address20 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0

Bit	Name	Description	Access	Reset
27	mbc_3	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address20 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0
26	mbc_2	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address20 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0

Bit	Name	Description	Access	Reset
25	mbc_1	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address20 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0
24	mbc_0	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address20 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0
23:16	reserved_23_16	Reserved	RO	0x0
15:0	addrhi	MAC Address20 [47:32] This field contains the upper 16 bits (47:32) of the 20th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address20_low

Register 521 (MAC Address20 Low Register)

The MAC Address20 Low register holds the lower 32 bits of the 21st 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800824
i_emac_emac1	0xFF802000	0xFF802824
i_emac_emac2	0xFF804000	0xFF804824

Offset: 0x824

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address20_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address20 [31:0] This field contains the lower 32 bits of the 21st 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address21_high

Register 522 (MAC Address21 High Register)

The MAC Address21 High register holds the upper 16 bits of the 22nd 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address21 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address21 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800828
i_emac_emac1	0xFF802000	0xFF802828
i_emac_emac2	0xFF804000	0xFF804828

Offset: 0x828

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address21_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 22nd MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
30	sa	Source Address When this bit is set, the MAC Address21[47:0] is used to compare with the SA fields of the received frame. When this bit is reset, the MAC Address21[47:0] is used to compare with the DA fields of the received frame. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
29	mbc_5	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address21 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0
28	mbc_4	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address21 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0

Bit	Name	Description	Access	Reset
27	mbc_3	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address21 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0
26	mbc_2	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address21 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0

Bit	Name	Description	Access	Reset
25	mbc_1	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address21 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0
24	mbc_0	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address21 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0
23:16	reserved_23_16	Reserved	RO	0x0
15:0	addrhi	MAC Address21 [47:32] This field contains the upper 16 bits (47:32) of the 6-byte 22nd MAC address.	RW	0xFFFF

gmacgrp_mac_address21_low

Register 523 (MAC Address21 Low Register)

The MAC Address21 Low register holds the lower 32 bits of the 22nd 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF80082C
i_emac_emac1	0xFF802000	0xFF80282C
i_emac_emac2	0xFF804000	0xFF80482C

Offset: 0x82C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address21_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address21 [31:0] This field contains the lower 32 bits of the 22nd 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address22_high

Register 524 (MAC Address22 High Register)

The MAC Address22 High register holds the upper 16 bits of the 23rd 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address22 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address22 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800830
i_emac_emac1	0xFF802000	0xFF802830
i_emac_emac2	0xFF804000	0xFF804830

Offset: 0x830

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address22_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 23rd MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30	sa	Source Address When this bit is set, the MAC Address22[47:0] is used to compare with the SA fields of the received frame. When this bit is reset, the MAC Address22[47:0] is used to compare with the DA fields of the received frame. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
29	mbc_5	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address22 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0
28	mbc_4	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address22 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0

Bit	Name	Description	Access	Reset
27	mbc_3	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address22 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0
26	mbc_2	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address22 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0

Bit	Name	Description	Access	Reset
25	mbc_1	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address22 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0
24	mbc_0	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address22 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0
23:16	reserved_23_16	Reserved	RO	0x0
15:0	addrhi	MAC Address22 [47:32] This field contains the upper 16 bits (47:32) of the 23rd 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address22_low

Register 525 (MAC Address22 Low Register)

The MAC Address22 Low register holds the lower 32 bits of the 23rd 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800834
i_emac_emac1	0xFF802000	0xFF802834
i_emac_emac2	0xFF804000	0xFF804834

Offset: 0x834

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address22_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address22 [31:0] This field contains the lower 32 bits of the 23rd 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address23_high

Register 526 (MAC Address23 High Register

The MAC Address23 High register holds the upper 16 bits of the 24th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address23 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address23 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800838
i_emac_emac1	0xFF802000	0xFF802838
i_emac_emac2	0xFF804000	0xFF804838

Offset: 0x838

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address23_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 24th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30	sa	Source Address When this bit is set, the MAC Address23[47:0] is used to compare with the SA fields of the received frame. When this bit is reset, the MAC Address23[47:0] is used to compare with the DA fields of the received frame. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
29	mbc_5	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address23 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0
28	mbc_4	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address23 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0

Bit	Name	Description	Access	Reset
27	mbc_3	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address23 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0
26	mbc_2	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address23 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0

Bit	Name	Description	Access	Reset
25	mbc_1	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address23 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0
24	mbc_0	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address23 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0
23:16	reserved_23_16	Reserved	RO	0x0
15:0	addrhi	MAC Address23 [47:32] This field contains the upper 16 bits (47:32) of the 24th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address23_low

Register 527 (MAC Address23 Low Register)

The MAC Address23 Low register holds the lower 32 bits of the 24th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF80083C
i_emac_emac1	0xFF802000	0xFF80283C
i_emac_emac2	0xFF804000	0xFF80483C

Offset: 0x83C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address23_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address23 [31:0] This field contains the lower 32 bits of the 24th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address24_high

Register 528 (MAC Address24 High Register)

The MAC Address24 High register holds the upper 16 bits of the 25th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address24 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address24 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800840
i_emac_emac1	0xFF802000	0xFF802840
i_emac_emac2	0xFF804000	0xFF804840

Offset: 0x840

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address24_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 25th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30	sa	Source Address When this bit is set, the MAC Address24[47:0] is used to compare with the SA fields of the received frame. When this bit is reset, the MAC Address24[47:0] is used to compare with the DA fields of the received frame. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
29	mbc_5	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address24 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0
28	mbc_4	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address24 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0

Bit	Name	Description	Access	Reset
27	mbc_3	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address24 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0
26	mbc_2	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address24 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0

Bit	Name	Description	Access	Reset
25	mbc_1	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address24 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0
24	mbc_0	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address24 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0
23:16	reserved_23_16	Reserved	RO	0x0
15:0	addrhi	MAC Address1 [47:32] This field contains the upper 16 bits (47:32) of the 25th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address24_low

Register 529 (MAC Address24 Low Register)

The MAC Address24 Low register holds the lower 32 bits of the 25th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800844
i_emac_emac1	0xFF802000	0xFF802844
i_emac_emac2	0xFF804000	0xFF804844

Offset: 0x844

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address24_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address24 [31:0] This field contains the lower 32 bits of the 25th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address25_high

Register 530 (MAC Address25 High Register)

The MAC Address25 High register holds the upper 16 bits of the 6-byte 26th MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address25 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address25 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800848
i_emac_emac1	0xFF802000	0xFF802848
i_emac_emac2	0xFF804000	0xFF804848

Offset: 0x848

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address25_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 26th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30	sa	Source Address When this bit is set, the MAC Address25[47:0] is used to compare with the SA fields of the received frame. When this bit is reset, the MAC Address25[47:0] is used to compare with the DA fields of the received frame. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
29	mbc_5	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address25 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0
28	mbc_4	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address25 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0

Bit	Name	Description	Access	Reset
27	mbc_3	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address25 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0
26	mbc_2	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address25 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0

Bit	Name	Description	Access	Reset
25	mbc_1	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address25 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0
24	mbc_0	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address25 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0
23:16	reserved_23_16	Reserved	RO	0x0
15:0	addrhi	MAC Address25 [47:32] This field contains the upper 16 bits (47:32) of the 26th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address25_low

Register 531 (MAC Address25 Low Register)

The MAC Address25 Low register holds the lower 32 bits of the 26th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF80084C
i_emac_emac1	0xFF802000	0xFF80284C
i_emac_emac2	0xFF804000	0xFF80484C

Offset: 0x84C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address25_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address25 [31:0] This field contains the lower 32 bits of the 26th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address26_high

Register 532 (MAC Address26 High Register)

The MAC Address26 High register holds the upper 16 bits of the 27th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address26 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address26 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800850
i_emac_emac1	0xFF802000	0xFF802850
i_emac_emac2	0xFF804000	0xFF804850

Offset: 0x850

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address26_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 27th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30	sa	Source Address When this bit is set, the MAC Address26[47:0] is used to compare with the SA fields of the received frame. When this bit is reset, the MAC Address26[47:0] is used to compare with the DA fields of the received frame. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
29	mbc_5	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address26 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0
28	mbc_4	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address26 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0

Bit	Name	Description	Access	Reset
27	mbc_3	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address26 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0
26	mbc_2	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address26 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0

Bit	Name	Description	Access	Reset
25	mbc_1	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address26 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0
24	mbc_0	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address26 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0
23:16	reserved_23_16	Reserved	RO	0x0
15:0	addrhi	MAC Address26 [47:32] This field contains the upper 16 bits (47:32) of the 27th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address26_low

Register 533 (MAC Address26 Low Register)

The MAC Address26 Low register holds the lower 32 bits of the 27th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800854
i_emac_emac1	0xFF802000	0xFF802854
i_emac_emac2	0xFF804000	0xFF804854

Offset: 0x854

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address26_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address26 [31:0] This field contains the lower 32 bits of the 27th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address27_high

Register 534 (MAC Address27 High Register)

The MAC Address27 High register holds the upper 16 bits of the 28th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address27 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address27 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800858
i_emac_emac1	0xFF802000	0xFF802858
i_emac_emac2	0xFF804000	0xFF804858

Offset: 0x858

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address27_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 28th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30	sa	Source Address When this bit is set, the MAC Address27[47:0] is used to compare with the SA fields of the received frame. When this bit is reset, the MAC Address27[47:0] is used to compare with the DA fields of the received frame. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
29	mbc_5	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address27 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0
28	mbc_4	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address27 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0

Bit	Name	Description	Access	Reset
27	mbc_3	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address27 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0
26	mbc_2	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address27 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0

Bit	Name	Description	Access	Reset
25	mbc_1	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address27 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0
24	mbc_0	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address27 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0
23:16	reserved_23_16	Reserved	RO	0x0
15:0	addrhi	MAC Address27 [47:32] This field contains the upper 16 bits (47:32) of the 28th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address27_low

Register 535 (MAC Address27 Low Register)

The MAC Address27 Low register holds the lower 32 bits of the 28th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF80085C
i_emac_emac1	0xFF802000	0xFF80285C
i_emac_emac2	0xFF804000	0xFF80485C

Offset: 0x85C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address27_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address27 [31:0] This field contains the lower 32 bits of the 28th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address28_high

Register 536 (MAC Address28 High Register)

The MAC Address28 High register holds the upper 16 bits of the 29th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address28 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address28 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800860
i_emac_emac1	0xFF802000	0xFF802860
i_emac_emac2	0xFF804000	0xFF804860

Offset: 0x860

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address28_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 29th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30	sa	Source Address When this bit is set, the MAC Address28[47:0] is used to compare with the SA fields of the received frame. When this bit is reset, the MAC Address28[47:0] is used to compare with the DA fields of the received frame. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
29	mbc_5	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address28 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0
28	mbc_4	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address28 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0

Bit	Name	Description	Access	Reset
27	mbc_3	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address28 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0
26	mbc_2	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address28 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0

Bit	Name	Description	Access	Reset
25	mbc_1	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address28 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0
24	mbc_0	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address28 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0
23:16	reserved_23_16	Reserved	RO	0x0
15:0	addrhi	MAC Address28 [47:32] This field contains the upper 16 bits (47:32) of the 29th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address28_low

Register 537 (MAC Address28 Low Register)

The MAC Address28 Low register holds the lower 32 bits of the 29th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800864
i_emac_emac1	0xFF802000	0xFF802864
i_emac_emac2	0xFF804000	0xFF804864

Offset: 0x864

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address28_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address28 [31:0] This field contains the lower 32 bits of the 29th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address29_high

Register 538 (MAC Address29 High Register)

The MAC Address29 High register holds the upper 16 bits of the 6-byte 30th MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address29 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address29 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800868
i_emac_emac1	0xFF802000	0xFF802868
i_emac_emac2	0xFF804000	0xFF804868

Offset: 0x868

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address29_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 30th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30	sa	Source Address When this bit is set, the MAC Address29[47:0] is used to compare with the SA fields of the received frame. When this bit is reset, the MAC Address29[47:0] is used to compare with the DA fields of the received frame. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
29	mbc_5	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address29 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0
28	mbc_4	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address29 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0

Bit	Name	Description	Access	Reset
27	mbc_3	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address29 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0
26	mbc_2	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address29 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0

Bit	Name	Description	Access	Reset
25	mbc_1	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address29 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0
24	mbc_0	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address29 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0
23:16	reserved_23_16	Reserved	RO	0x0
15:0	addrhi	MAC Address29 [47:32] This field contains the upper 16 bits (47:32) of the 30th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address29_low

Register 539 (MAC Address29 Low Register)

The MAC Address29 Low register holds the lower 32 bits of the 30th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF80086C
i_emac_emac1	0xFF802000	0xFF80286C
i_emac_emac2	0xFF804000	0xFF80486C

Offset: 0x86C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address29_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address29 [31:0] This field contains the lower 32 bits of the 30th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address30_high

Register 540 (MAC Address30 High Register)

The MAC Address30 High register holds the upper 16 bits of the 31st 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address30 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address30 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800870
i_emac_emac1	0xFF802000	0xFF802870
i_emac_emac2	0xFF804000	0xFF804870

Offset: 0x870

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address30_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 31st MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
30	sa	Source Address When this bit is set, the MAC Address30[47:0] is used to compare with the SA fields of the received frame. When this bit is reset, the MAC Address30[47:0] is used to compare with the DA fields of the received frame. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
29	mbc_5	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address30 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0
28	mbc_4	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address30 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0

Bit	Name	Description	Access	Reset
27	mbc_3	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address30 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0
26	mbc_2	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address30 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0

Bit	Name	Description	Access	Reset
25	mbc_1	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address30 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0
24	mbc_0	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address30 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0
23:16	reserved_23_16	Reserved	RO	0x0
15:0	addrhi	MAC Address30 [47:32] This field contains the upper 16 bits (47:32) of the 31st 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address30_low

Register 541 (MAC Address30 Low Register)

The MAC Address30 Low register holds the lower 32 bits of the 31st 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800874
i_emac_emac1	0xFF802000	0xFF802874
i_emac_emac2	0xFF804000	0xFF804874

Offset: 0x874

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address30_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address30 [31:0] This field contains the lower 32 bits of the 31st 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address31_high

Register 542 (MAC Address31 High Register)

The MAC Address31 High register holds the upper 16 bits of the 32nd 6-byte MAC address of the station. If the MAC address registers are

configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address31 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address31 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800878
i_emac_emac1	0xFF802000	0xFF802878
i_emac_emac2	0xFF804000	0xFF804878

Offset: 0x878

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	sa RW 0x0	mbc_5 RW 0x0	mbc_4 RW 0x0	mbc_3 RW 0x0	mbc_2 RW 0x0	mbc_1 RW 0x0	mbc_0 RW 0x0	reserved_23_16 RO 0x0							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address31_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 32nd MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
30	sa	Source Address When this bit is set, the MAC Address31[47:0] is used to compare with the SA fields of the received frame. When this bit is reset, the MAC Address31[47:0] is used to compare with the DA fields of the received frame. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
29	mbc_5	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address31 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** **Description** 0x0 UNMASKED 0x1 MASKED	RW	0x0

Bit	Name	Description	Access	Reset
28	mbc_4	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address31 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0
27	mbc_3	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address31 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0

Bit	Name	Description	Access	Reset
26	mbc_2	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address31 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0
25	mbc_1	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address31 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). **Value** — **Description** 0x0 — UNMASKED 0x1 — MASKED	RW	0x0

Bit	Name	Description	Access	Reset
24	mbc_0	This array of bits are mask control bits for comparison of each of the MAC Address bytes. When masked, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address31 high and low registers. Each bit controls the masking of the bytes. You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. The array index corresponds to the byte (e.g. index 0 is for bits 7:0). Value: Description 0x0: UNMASKED 0x1: MASKED	RW	0x0
23:16	reserved_23_16	Reserved	RO	0x0
15:0	addrhi	MAC Address31 [47:32] This field contains the upper 16 bits (47:32) of the 32nd 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address31_low

Register 543 (MAC Address31 Low Register)

The MAC Address31 Low register holds the lower 32 bits of the 32nd 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF80087C
i_emac_emac1	0xFF802000	0xFF80287C
i_emac_emac2	0xFF804000	0xFF80487C

Offset: 0x87C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address31_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address31 [31:0] This field contains the lower 32 bits of the 32nd 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address32_high

Register 544 (MAC Address32 High Register)

The MAC Address32 High register holds the upper 16 bits of the 33rd 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address32 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address32 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800880
i_emac_emac1	0xFF802000	0xFF802880
i_emac_emac2	0xFF804000	0xFF804880

Offset: 0x880

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address32_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 33rd MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address32 [47:32] This field contains the upper 16 bits (47:32) of the 33rd 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address32_low

Register 545 (MAC Address32 Low Register)

The MAC Address32 Low register holds the lower 32 bits of the 33rd 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800884
i_emac_emac1	0xFF802000	0xFF802884
i_emac_emac2	0xFF804000	0xFF804884

Offset: 0x884

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address32_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address32 [31:0] This field contains the lower 32 bits of the 33rd 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address33_high

Register 546 (MAC Address33 High Register)

The MAC Address33 High register holds the upper 16 bits of the 34th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address33 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address33 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800888
i_emac_emac1	0xFF802000	0xFF802888
i_emac_emac2	0xFF804000	0xFF804888

Offset: 0x888

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address33_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 34th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address33 [47:32] This field contains the upper 16 bits (47:32) of the 34th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address33_low

Register 547 (MAC Address33 Low Register)

The MAC Address33 Low register holds the lower 32 bits of the 34th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF80088C
i_emac_emac1	0xFF802000	0xFF80288C
i_emac_emac2	0xFF804000	0xFF80488C

Offset: 0x88C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address33_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address33 [31:0] This field contains the lower 32 bits of the 34th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address34_high

Register 548 (MAC Address34 High Register)

The MAC Address34 High register holds the upper 16 bits of the 35th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address34 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address34 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800890
i_emac_emac1	0xFF802000	0xFF802890
i_emac_emac2	0xFF804000	0xFF804890

Offset: 0x890

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address34_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 35th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address34 [47:32] This field contains the upper 16 bits (47:32) of the 35th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address34_low

Register 549 (MAC Address34 Low Register)

The MAC Address34 Low register holds the lower 32 bits of the 35th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800894
i_emac_emac1	0xFF802000	0xFF802894
i_emac_emac2	0xFF804000	0xFF804894

Offset: 0x894

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address34_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address34 [31:0] This field contains the lower 32 bits of the 35th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address35_high

Register 550 (MAC Address35 High Register)

The MAC Address35 High register holds the upper 16 bits of the 36th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address35 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address35 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800898
i_emac_emac1	0xFF802000	0xFF802898
i_emac_emac2	0xFF804000	0xFF804898

Offset: 0x898

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address35_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 36th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address35 [47:32] This field contains the upper 16 bits (47:32) of the 36th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address35_low

Register 551 (MAC Address35 Low Register)

The MAC Address35 Low register holds the lower 32 bits of the 36th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF80089C
i_emac_emac1	0xFF802000	0xFF80289C
i_emac_emac2	0xFF804000	0xFF80489C

Offset: 0x89C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address35_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address35 [31:0] This field contains the lower 32 bits of the 36th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address36_high

Register 552 (MAC Address36 High Register)

The MAC Address36 High register holds the upper 16 bits of the 37th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address36 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address36 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8008A0
i_emac_emac1	0xFF802000	0xFF8028A0
i_emac_emac2	0xFF804000	0xFF8048A0

Offset: 0x8A0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address36_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 37th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address36 [47:32] This field contains the upper 16 bits (47:32) of the 37th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address36_low

Register 553 (MAC Address36 Low Register)

The MAC Address36 Low register holds the lower 32 bits of the 34th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8008A4
i_emac_emac1	0xFF802000	0xFF8028A4
i_emac_emac2	0xFF804000	0xFF8048A4

Offset: 0x8A4

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address36_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address36 [31:0] This field contains the lower 32 bits of the 37th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address37_high

Register 554 (MAC Address37 High Register)

The MAC Address37 High register holds the upper 16 bits of the 38th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address37 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address37 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8008A8
i_emac_emac1	0xFF802000	0xFF8028A8
i_emac_emac2	0xFF804000	0xFF8048A8

Offset: 0x8A8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address37_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 38th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. Value: Description 0x0: DISABLED 0x1: ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address37 [47:32] This field contains the upper 16 bits (47:32) of the 38th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address37_low

Register 555 (MAC Address37 Low Register)

The MAC Address37 Low register holds the lower 32 bits of the 37th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8008AC
i_emac_emac1	0xFF802000	0xFF8028AC
i_emac_emac2	0xFF804000	0xFF8048AC

Offset: 0x8AC

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address37_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address37 [31:0] This field contains the lower 32 bits of the 38th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address38_high

Register 556 (MAC Address38 High Register)

The MAC Address38 High register holds the upper 16 bits of the 39th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address38 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address38 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8008B0
i_emac_emac1	0xFF802000	0xFF8028B0
i_emac_emac2	0xFF804000	0xFF8048B0

Offset: 0x8B0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address38_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 39th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address38 [47:32] This field contains the upper 16 bits (47:32) of the 39th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address38_low

Register 557 (MAC Address38 Low Register)

The MAC Address38 Low register holds the lower 32 bits of the 39th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8008B4
i_emac_emac1	0xFF802000	0xFF8028B4
i_emac_emac2	0xFF804000	0xFF8048B4

Offset: 0x8B4

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address38_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address38 [31:0] This field contains the lower 32 bits of the 39th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address39_high

Register 558 (MAC Address39 High Register)

The MAC Address39 High register holds the upper 16 bits of the 40th 6-byte MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address40 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address40 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8008B8
i_emac_emac1	0xFF802000	0xFF8028B8
i_emac_emac2	0xFF804000	0xFF8048B8

Offset: 0x8B8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address39_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 40th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address39 [47:32] This field contains the upper 16 bits (47:32) of the 40th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address39_low

Register 559 (MAC Address39 Low Register)

The MAC Address39 Low register holds the lower 32 bits of the 40th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8008BC
i_emac_emac1	0xFF802000	0xFF8028BC
i_emac_emac2	0xFF804000	0xFF8048BC

Offset: 0x8BC

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address39_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address39 [31:0] This field contains the lower 32 bits of the 40th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address40_high

Register 560 (MAC Address40 High Register)

The MAC Address40 High register holds the upper 16 bits of the 41st 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address40 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address40 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8008C0
i_emac_emac1	0xFF802000	0xFF8028C0
i_emac_emac2	0xFF804000	0xFF8048C0

Offset: 0x8C0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address40_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 41st MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address40 [47:32] This field contains the upper 16 bits (47:32) of the 41st 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address40_low

Register 561 (MAC Address40 Low Register)

The MAC Address40 Low register holds the lower 32 bits of the 41st 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8008C4
i_emac_emac1	0xFF802000	0xFF8028C4
i_emac_emac2	0xFF804000	0xFF8048C4

Offset: 0x8C4

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address40_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address40 [31:0] This field contains the lower 32 bits of the 41st 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address41_high

Register 562 (MAC Address41 High Register)

The MAC Address41 High register holds the upper 16 bits of the 42nd 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address41 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address41 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8008C8
i_emac_emac1	0xFF802000	0xFF8028C8
i_emac_emac2	0xFF804000	0xFF8048C8

Offset: 0x8C8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address41_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 42nd MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address41 [47:32] This field contains the upper 16 bits (47:32) of the 42nd 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address41_low

Register 563 (MAC Address41 Low Register)

The MAC Address41 Low register holds the lower 32 bits of the 42nd 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8008CC
i_emac_emac1	0xFF802000	0xFF8028CC
i_emac_emac2	0xFF804000	0xFF8048CC

Offset: 0x8CC

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address41_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address41 [31:0] This field contains the lower 32 bits of the 42nd 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address42_high

Register 564 (MAC Address42 High Register)

The MAC Address42 High register holds the upper 16 bits of the 43rd 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address42 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address42 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8008D0
i_emac_emac1	0xFF802000	0xFF8028D0
i_emac_emac2	0xFF804000	0xFF8048D0

Offset: 0x8D0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address42_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 43rd MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address42 [47:32] This field contains the upper 16 bits (47:32) of the 43rd 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address42_low

Register 565 (MAC Address42 Low Register)

The MAC Address42 Low register holds the lower 32 bits of the 43rd 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8008D4
i_emac_emac1	0xFF802000	0xFF8028D4
i_emac_emac2	0xFF804000	0xFF8048D4

Offset: 0x8D4

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address42_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address42 [31:0] This field contains the lower 32 bits of the 43rd 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address43_high

Register 566 (MAC Address43 High Register)

The MAC Address43 High register holds the upper 16 bits of the 44th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address43 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address43 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8008D8
i_emac_emac1	0xFF802000	0xFF8028D8
i_emac_emac2	0xFF804000	0xFF8048D8

Offset: 0x8D8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address43_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 44th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. Value: Description 0x0: DISABLED 0x1: ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address43 [47:32] This field contains the upper 16 bits (47:32) of the 44th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address43_low

Register 567 (MAC Address43 Low Register)

The MAC Address43 Low register holds the lower 32 bits of the 44th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8008DC
i_emac_emac1	0xFF802000	0xFF8028DC
i_emac_emac2	0xFF804000	0xFF8048DC

Offset: 0x8DC

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address43_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address43 [31:0] This field contains the lower 32 bits of the 44th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address44_high

Register 568 (MAC Address44 High Register)

The MAC Address44 High register holds the upper 16 bits of the 45th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address44 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address44 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8008E0
i_emac_emac1	0xFF802000	0xFF8028E0
i_emac_emac2	0xFF804000	0xFF8048E0

Offset: 0x8E0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address44_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 45th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address44 [47:32] This field contains the upper 16 bits (47:32) of the 45th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address44_low

Register 569 (MAC Address44 Low Register)

The MAC Address44 Low register holds the lower 32 bits of the 45th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8008E4
i_emac_emac1	0xFF802000	0xFF8028E4
i_emac_emac2	0xFF804000	0xFF8048E4

Offset: 0x8E4

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address44_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address44 [31:0] This field contains the lower 32 bits of the 45th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address45_high

Register 570 (MAC Address45 High Register)

The MAC Address45 High register holds the upper 16 bits of the 46th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address45 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address45 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8008E8
i_emac_emac1	0xFF802000	0xFF8028E8
i_emac_emac2	0xFF804000	0xFF8048E8

Offset: 0x8E8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address45_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 46th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address45 [47:32] This field contains the upper 16 bits (47:32) of the 46th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address45_low

Register 571 (MAC Address45 Low Register)

The MAC Address45 Low register holds the lower 32 bits of the 46th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8008EC
i_emac_emac1	0xFF802000	0xFF8028EC
i_emac_emac2	0xFF804000	0xFF8048EC

Offset: 0x8EC

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address45_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address45 [31:0] This field contains the lower 32 bits of the 46th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address46_high

Register 572 (MAC Address46 High Register)

The MAC Address46 High register holds the upper 16 bits of the 47th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address46 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address46 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8008F0
i_emac_emac1	0xFF802000	0xFF8028F0
i_emac_emac2	0xFF804000	0xFF8048F0

Offset: 0x8F0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address46_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 47th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. Value: Description 0x0: DISABLED 0x1: ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address46 [47:32] This field contains the upper 16 bits (47:32) of the 47th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address46_low

Register 573 (MAC Address46 Low Register)

The MAC Address46 Low register holds the lower 32 bits of the 47th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8008F4
i_emac_emac1	0xFF802000	0xFF8028F4
i_emac_emac2	0xFF804000	0xFF8048F4

Offset: 0x8F4

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address46_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address46 [31:0] This field contains the lower 32 bits of the 47th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address47_high

Register 574 (MAC Address47 High Register)

The MAC Address47 High register holds the upper 16 bits of the 48th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address47 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address47 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8008F8
i_emac_emac1	0xFF802000	0xFF8028F8
i_emac_emac2	0xFF804000	0xFF8048F8

Offset: 0x8F8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address47_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 48th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address47 [47:32] This field contains the upper 16 bits (47:32) of the 48th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address47_low

Register 575 (MAC Address47 Low Register)

The MAC Address47 Low register holds the lower 32 bits of the 48th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8008FC
i_emac_emac1	0xFF802000	0xFF8028FC
i_emac_emac2	0xFF804000	0xFF8048FC

Offset: 0x8FC

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address47_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address47 [31:0] This field contains the lower 32 bits of the 48th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address48_high

Register 576 (MAC Address48 High Register)

The MAC Address48 High register holds the upper 16 bits of the 49th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address48 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address48 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800900
i_emac_emac1	0xFF802000	0xFF802900
i_emac_emac2	0xFF804000	0xFF804900

Offset: 0x900

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address48_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 49th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address48 [47:32] This field contains the upper 16 bits (47:32) of the 49th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address48_low

Register 577 (MAC Address48 Low Register)

The MAC Address48 Low register holds the lower 32 bits of the 49th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800904
i_emac_emac1	0xFF802000	0xFF802904
i_emac_emac2	0xFF804000	0xFF804904

Offset: 0x904

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address48_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address48 [31:0] This field contains the lower 32 bits of the 49th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address49_high

Register 578 (MAC Address49 High Register)

The MAC Address49 High register holds the upper 16 bits of the 50th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address49 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address49 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800908
i_emac_emac1	0xFF802000	0xFF802908
i_emac_emac2	0xFF804000	0xFF804908

Offset: 0x908

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address49_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 50th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. Value: Description 0x0: DISABLED 0x1: ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address49 [47:32] This field contains the upper 16 bits (47:32) of the 50th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address49_low

Register 579 (MAC Address49 Low Register)

The MAC Address49 Low register holds the lower 32 bits of the 50th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF80090C
i_emac_emac1	0xFF802000	0xFF80290C
i_emac_emac2	0xFF804000	0xFF80490C

Offset: 0x90C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address49_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address49 [31:0] This field contains the lower 32 bits of the 50th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address50_high

Register 580 (MAC Address50 High Register)

The MAC Address50 High register holds the upper 16 bits of the 51st 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address50 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address50 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800910
i_emac_emac1	0xFF802000	0xFF802910
i_emac_emac2	0xFF804000	0xFF804910

Offset: 0x910

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address50_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 51st MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address50 [47:32] This field contains the upper 16 bits (47:32) of the 51st 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address50_low

Register 581 (MAC Address50 Low Register)

The MAC Address50 Low register holds the lower 32 bits of the 51st 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800914
i_emac_emac1	0xFF802000	0xFF802914
i_emac_emac2	0xFF804000	0xFF804914

Offset: 0x914

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address50_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address50 [31:0] This field contains the lower 32 bits of the 51st 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address51_high

Register 582 (MAC Address51 High Register)

The MAC Address51 High register holds the upper 16 bits of the 52nd 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address51 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address51 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800918
i_emac_emac1	0xFF802000	0xFF802918
i_emac_emac2	0xFF804000	0xFF804918

Offset: 0x918

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address51_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 52nd MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address51 [47:32] This field contains the upper 16 bits (47:32) of the 52nd 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address51_low

Register 583 (MAC Address51 Low Register)

The MAC Address51 Low register holds the lower 32 bits of the 52nd 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF80091C
i_emac_emac1	0xFF802000	0xFF80291C
i_emac_emac2	0xFF804000	0xFF80491C

Offset: 0x91C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address51_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address51 [31:0] This field contains the lower 32 bits of the 52nd 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address52_high

Register 584 (MAC Address52 High Register)

The MAC Address52 High register holds the upper 16 bits of the 53rd 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address52 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address52 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800920
i_emac_emac1	0xFF802000	0xFF802920
i_emac_emac2	0xFF804000	0xFF804920

Offset: 0x920

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address52_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 53rd MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address52 [47:32] This field contains the upper 16 bits (47:32) of the 53rd 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address52_low

Register 585 (MAC Address52 Low Register)

The MAC Address52 Low register holds the lower 32 bits of the 53rd 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800924
i_emac_emac1	0xFF802000	0xFF802924
i_emac_emac2	0xFF804000	0xFF804924

Offset: 0x924

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address52_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address52 [31:0] This field contains the lower 32 bits of the 53rd 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address53_high

Register 586 (MAC Address53 High Register)

The MAC Address53 High register holds the upper 16 bits of the 54th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address53 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address53 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800928
i_emac_emac1	0xFF802000	0xFF802928
i_emac_emac2	0xFF804000	0xFF804928

Offset: 0x928

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address53_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 54th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address53 [47:32] This field contains the upper 16 bits (47:32) of the 54th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address53_low

Register 587 (MAC Address53 Low Register)

The MAC Address53 Low register holds the lower 32 bits of the 54th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF80092C
i_emac_emac1	0xFF802000	0xFF80292C
i_emac_emac2	0xFF804000	0xFF80492C

Offset: 0x92C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address53_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address53 [31:0] This field contains the lower 32 bits of the 54th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address54_high

Register 588 (MAC Address54 High Register)

The MAC Address54 High register holds the upper 16 bits of the 55th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address54 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address54 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800930
i_emac_emac1	0xFF802000	0xFF802930
i_emac_emac2	0xFF804000	0xFF804930

Offset: 0x930

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address54_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 55th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address54 [47:32] This field contains the upper 16 bits (47:32) of the 55th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address54_low

Register 589 (MAC Address54 Low Register)

The MAC Address54 Low register holds the lower 32 bits of the 55th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800934
i_emac_emac1	0xFF802000	0xFF802934
i_emac_emac2	0xFF804000	0xFF804934

Offset: 0x934

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address54_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address54 [31:0] This field contains the lower 32 bits of the 55th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address55_high

Register 590 (MAC Address55 High Register)

The MAC Address55 High register holds the upper 16 bits of the 56th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address55 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address55 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800938
i_emac_emac1	0xFF802000	0xFF802938
i_emac_emac2	0xFF804000	0xFF804938

Offset: 0x938

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address55_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 56th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. Value: Description 0x0: DISABLED 0x1: ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address55 [47:32] This field contains the upper 16 bits (47:32) of the 56th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address55_low

Register 591 (MAC Address55 Low Register)

The MAC Address55 Low register holds the lower 32 bits of the 56th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF80093C
i_emac_emac1	0xFF802000	0xFF80293C
i_emac_emac2	0xFF804000	0xFF80493C

Offset: 0x93C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address55_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address55 [31:0] This field contains the lower 32 bits of the 56th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address56_high

Register 592 (MAC Address56 High Register)

The MAC Address56 High register holds the upper 16 bits of the 57th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address56 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address56 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800940
i_emac_emac1	0xFF802000	0xFF802940
i_emac_emac2	0xFF804000	0xFF804940

Offset: 0x940

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address56_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 57th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address56 [47:32] This field contains the upper 16 bits (47:32) of the 57th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address56_low

Register 593 (MAC Address56 Low Register)

The MAC Address56 Low register holds the lower 32 bits of the 57th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800944
i_emac_emac1	0xFF802000	0xFF802944
i_emac_emac2	0xFF804000	0xFF804944

Offset: 0x944

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address56_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address56 [31:0] This field contains the lower 32 bits of the 57th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address57_high

Register 594 (MAC Address57 High Register)

The MAC Address57 High register holds the upper 16 bits of the 58th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address57 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address57 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800948
i_emac_emac1	0xFF802000	0xFF802948
i_emac_emac2	0xFF804000	0xFF804948

Offset: 0x948

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address57_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 58th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address57 [47:32] This field contains the upper 16 bits (47:32) of the 58th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address57_low

Register 595 (MAC Address57 Low Register)

The MAC Address57 Low register holds the lower 32 bits of the 58th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF80094C
i_emac_emac1	0xFF802000	0xFF80294C
i_emac_emac2	0xFF804000	0xFF80494C

Offset: 0x94C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address57_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address57 [31:0] This field contains the lower 32 bits of the 58th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address58_high

Register 596 (MAC Address58 High Register)

The MAC Address58 High register holds the upper 16 bits of the 59th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address58 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address58 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800950
i_emac_emac1	0xFF802000	0xFF802950
i_emac_emac2	0xFF804000	0xFF804950

Offset: 0x950

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address58_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 59th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address58 [47:32] This field contains the upper 16 bits (47:32) of the 59th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address58_low

Register 597 (MAC Address58 Low Register)

The MAC Address58 Low register holds the lower 32 bits of the 59th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800954
i_emac_emac1	0xFF802000	0xFF802954
i_emac_emac2	0xFF804000	0xFF804954

Offset: 0x954

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address58_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address58 [31:0] This field contains the lower 32 bits of the 59th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address59_high

Register 598 (MAC Address59 High Register)

The MAC Address59 High register holds the upper 16 bits of the 60th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address59 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address59 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800958
i_emac_emac1	0xFF802000	0xFF802958
i_emac_emac2	0xFF804000	0xFF804958

Offset: 0x958

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address59_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 60th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address59 [47:32] This field contains the upper 16 bits (47:32) of the 60th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address59_low

Register 599 (MAC Address59 Low Register)

The MAC Address59 Low register holds the lower 32 bits of the 60th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF80095C
i_emac_emac1	0xFF802000	0xFF80295C
i_emac_emac2	0xFF804000	0xFF80495C

Offset: 0x95C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address59_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address59 [31:0] This field contains the lower 32 bits of the 60th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address60_high

Register 600 (MAC Address60 High Register)

The MAC Address60 High register holds the upper 16 bits of the 61st 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address60 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address60 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800960
i_emac_emac1	0xFF802000	0xFF802960
i_emac_emac2	0xFF804000	0xFF804960

Offset: 0x960

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address60_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 61st MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address60 [47:32] This field contains the upper 16 bits (47:32) of the 61st 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address60_low

Register 601 (MAC Address60 Low Register)

The MAC Address60 Low register holds the lower 32 bits of the 61st 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800964
i_emac_emac1	0xFF802000	0xFF802964
i_emac_emac2	0xFF804000	0xFF804964

Offset: 0x964

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address60_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address60 [31:0] This field contains the lower 32 bits of the 61st 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address61_high

Register 602 (MAC Address61 High Register)

The MAC Address61 High register holds the upper 16 bits of the 62nd 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address61 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address61 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800968
i_emac_emac1	0xFF802000	0xFF802968
i_emac_emac2	0xFF804000	0xFF804968

Offset: 0x968

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address61_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 62nd MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. Value: Description 0x0: DISABLED 0x1: ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address61 [47:32] This field contains the upper 16 bits (47:32) of the 62nd 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address61_low

Register 603 (MAC Address61 Low Register)

The MAC Address61 Low register holds the lower 32 bits of the 62nd 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF80096C
i_emac_emac1	0xFF802000	0xFF80296C
i_emac_emac2	0xFF804000	0xFF80496C

Offset: 0x96C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address61_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address61 [31:0] This field contains the lower 32 bits of the 62nd 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address62_high

Register 604 (MAC Address62 High Register)

The MAC Address62 High register holds the upper 16 bits of the 63rd 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address62 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address62 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800970
i_emac_emac1	0xFF802000	0xFF802970
i_emac_emac2	0xFF804000	0xFF804970

Offset: 0x970

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address62_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 63rd MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address62 [47:32] This field contains the upper 16 bits (47:32) of the 63rd 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address62_low

Register 605 (MAC Address62 Low Register)

The MAC Address62 Low register holds the lower 32 bits of the 63rd 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800974
i_emac_emac1	0xFF802000	0xFF802974
i_emac_emac2	0xFF804000	0xFF804974

Offset: 0x974

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address62_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address62 [31:0] This field contains the lower 32 bits of the 63rd 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address63_high

Register 606 (MAC Address63 High Register)

The MAC Address63 High register holds the upper 16 bits of the 64th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address63 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address63 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800978
i_emac_emac1	0xFF802000	0xFF802978
i_emac_emac2	0xFF804000	0xFF804978

Offset: 0x978

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address63_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 64th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address63 [47:32] This field contains the upper 16 bits (47:32) of the 64th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address63_low

Register 607 (MAC Address63 Low Register)

The MAC Address63 Low register holds the lower 32 bits of the 64th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF80097C
i_emac_emac1	0xFF802000	0xFF80297C
i_emac_emac2	0xFF804000	0xFF80497C

Offset: 0x97C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address63_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address63 [31:0] This field contains the lower 32 bits of the 64th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address64_high

Register 608 (MAC Address64 High Register)

The MAC Address64 High register holds the upper 16 bits of the 65th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address64 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address64 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800980
i_emac_emac1	0xFF802000	0xFF802980
i_emac_emac2	0xFF804000	0xFF804980

Offset: 0x980

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address64_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 65th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address64 [47:32] This field contains the upper 16 bits (47:32) of the 65th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address64_low

Register 609 (MAC Address64 Low Register)

The MAC Address64 Low register holds the lower 32 bits of the 65th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800984
i_emac_emac1	0xFF802000	0xFF802984
i_emac_emac2	0xFF804000	0xFF804984

Offset: 0x984

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address64_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address64 [31:0] This field contains the lower 32 bits of the 65th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address65_high

Register 610 (MAC Address65 High Register)

The MAC Address65 High register holds the upper 16 bits of the 66th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address65 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address65 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800988
i_emac_emac1	0xFF802000	0xFF802988
i_emac_emac2	0xFF804000	0xFF804988

Offset: 0x988

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address65_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 66th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address65 [47:32] This field contains the upper 16 bits (47:32) of the 66th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address65_low

Register 611 (MAC Address65 Low Register)

The MAC Address65 Low register holds the lower 32 bits of the 66th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF80098C
i_emac_emac1	0xFF802000	0xFF80298C
i_emac_emac2	0xFF804000	0xFF80498C

Offset: 0x98C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address65_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address65 [31:0] This field contains the lower 32 bits of the 66th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address66_high

Register 612 (MAC Address66 High Register)

The MAC Address66 High register holds the upper 16 bits of the 67th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address66 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address66 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800990
i_emac_emac1	0xFF802000	0xFF802990
i_emac_emac2	0xFF804000	0xFF804990

Offset: 0x990

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address66_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 67th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address66 [47:32] This field contains the upper 16 bits (47:32) of the 67th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address66_low

Register 613 (MAC Address66 Low Register)

The MAC Address66 Low register holds the lower 32 bits of the 67th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800994
i_emac_emac1	0xFF802000	0xFF802994
i_emac_emac2	0xFF804000	0xFF804994

Offset: 0x994

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address66_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address66 [31:0] This field contains the lower 32 bits of the 67th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address67_high

Register 614 (MAC Address67 High Register)

The MAC Address67 High register holds the upper 16 bits of the 68th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address67 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address67 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800998
i_emac_emac1	0xFF802000	0xFF802998
i_emac_emac2	0xFF804000	0xFF804998

Offset: 0x998

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address67_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 68th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address67 [47:32] This field contains the upper 16 bits (47:32) of the 68th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address67_low

Register 615 (MAC Address67 Low Register)

The MAC Address67 Low register holds the lower 32 bits of the 68th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF80099C
i_emac_emac1	0xFF802000	0xFF80299C
i_emac_emac2	0xFF804000	0xFF80499C

Offset: 0x99C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address67_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address67 [31:0] This field contains the lower 32 bits of the 68th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address68_high

Register 616 (MAC Address68 High Register)

The MAC Address68 High register holds the upper 16 bits of the 69th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address68 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address68 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8009A0
i_emac_emac1	0xFF802000	0xFF8029A0
i_emac_emac2	0xFF804000	0xFF8049A0

Offset: 0x9A0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address68_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 69th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address68 [47:32] This field contains the upper 16 bits (47:32) of the 69th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address68_low

Register 617 (MAC Address68 Low Register)

The MAC Address68 Low register holds the lower 32 bits of the 69th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8009A4
i_emac_emac1	0xFF802000	0xFF8029A4
i_emac_emac2	0xFF804000	0xFF8049A4

Offset: 0x9A4

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address68_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address68 [31:0] This field contains the lower 32 bits of the 69th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address69_high

Register 618 (MAC Address69 High Register)

The MAC Address69 High register holds the upper 16 bits of the 70th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address69 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address70 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8009A8
i_emac_emac1	0xFF802000	0xFF8029A8
i_emac_emac2	0xFF804000	0xFF8049A8

Offset: 0x9A8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address69_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 70th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address69 [47:32] This field contains the upper 16 bits (47:32) of the 70th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address69_low

Register 619 (MAC Address69 Low Register)

The MAC Address69 Low register holds the lower 32 bits of the 70th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8009AC
i_emac_emac1	0xFF802000	0xFF8029AC
i_emac_emac2	0xFF804000	0xFF8049AC

Offset: 0x9AC

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address69_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address69 [31:0] This field contains the lower 32 bits of the 70th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address70_high

Register 620 (MAC Address70 High Register)

The MAC Address70 High register holds the upper 16 bits of the 71st 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address70 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address70 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8009B0
i_emac_emac1	0xFF802000	0xFF8029B0
i_emac_emac2	0xFF804000	0xFF8049B0

Offset: 0x9B0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address70_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 71st MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address70 [47:32] This field contains the upper 16 bits (47:32) of the 71st 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address70_low

Register 621 (MAC Address70 Low Register)

The MAC Address70 Low register holds the lower 32 bits of the 71st 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8009B4
i_emac_emac1	0xFF802000	0xFF8029B4
i_emac_emac2	0xFF804000	0xFF8049B4

Offset: 0x9B4

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address70_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address70 [31:0] This field contains the lower 32 bits of the 71st 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address71_high

Register 622 (MAC Address71 High Register)

The MAC Address71 High register holds the upper 16 bits of the 72nd 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address71 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address71 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8009B8
i_emac_emac1	0xFF802000	0xFF8029B8
i_emac_emac2	0xFF804000	0xFF8049B8

Offset: 0x9B8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address71_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 72nd MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address71 [47:32] This field contains the upper 16 bits (47:32) of the 72nd 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address71_low

Register 623 (MAC Address71 Low Register)

The MAC Address71 Low register holds the lower 32 bits of the 72nd 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8009BC
i_emac_emac1	0xFF802000	0xFF8029BC
i_emac_emac2	0xFF804000	0xFF8049BC

Offset: 0x9BC

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address71_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address71 [31:0] This field contains the lower 32 bits of the 72nd 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address72_high

Register 624 (MAC Address72 High Register)

The MAC Address72 High register holds the upper 16 bits of the 73rd 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address72 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address72 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8009C0
i_emac_emac1	0xFF802000	0xFF8029C0
i_emac_emac2	0xFF804000	0xFF8049C0

Offset: 0x9C0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address72_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 73rd MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address72 [47:32] This field contains the upper 16 bits (47:32) of the 73rd 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address72_low

Register 625 (MAC Address72 Low Register)

The MAC Address72 Low register holds the lower 32 bits of the 73rd 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8009C4
i_emac_emac1	0xFF802000	0xFF8029C4
i_emac_emac2	0xFF804000	0xFF8049C4

Offset: 0x9C4

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address72_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address72 [31:0] This field contains the lower 32 bits of the 73rd 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address73_high

Register 626 (MAC Address73 High Register)

The MAC Address73 High register holds the upper 16 bits of the 74th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address73 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address73 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8009C8
i_emac_emac1	0xFF802000	0xFF8029C8
i_emac_emac2	0xFF804000	0xFF8049C8

Offset: 0x9C8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address73_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 74th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. Value: Description 0x0: DISABLED 0x1: ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address73 [47:32] This field contains the upper 16 bits (47:32) of the 74th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address73_low

Register 627 (MAC Address73 Low Register)

The MAC Address73 Low register holds the lower 32 bits of the 74th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8009CC
i_emac_emac1	0xFF802000	0xFF8029CC
i_emac_emac2	0xFF804000	0xFF8049CC

Offset: 0x9CC

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address73_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address73 [31:0] This field contains the lower 32 bits of the 74th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address74_high

Register 628 (MAC Address74 High Register)

The MAC Address74 High register holds the upper 16 bits of the 75th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address74 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address74 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8009D0
i_emac_emac1	0xFF802000	0xFF8029D0
i_emac_emac2	0xFF804000	0xFF8049D0

Offset: 0x9D0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address74_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 75th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address74 [47:32] This field contains the upper 16 bits (47:32) of the 75th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address74_low

Register 629 (MAC Address74 Low Register)

The MAC Address74 Low register holds the lower 32 bits of the 75th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8009D4
i_emac_emac1	0xFF802000	0xFF8029D4
i_emac_emac2	0xFF804000	0xFF8049D4

Offset: 0x9D4

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address74_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address74 [31:0] This field contains the lower 32 bits of the 75th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address75_high

Register 630 (MAC Address75 High Register)

The MAC Address75 High register holds the upper 16 bits of the 76th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address75 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address75 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8009D8
i_emac_emac1	0xFF802000	0xFF8029D8
i_emac_emac2	0xFF804000	0xFF8049D8

Offset: 0x9D8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address75_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 76th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address75 [47:32] This field contains the upper 16 bits (47:32) of the 76th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address75_low

Register 631 (MAC Address75 Low Register)

The MAC Address75 Low register holds the lower 32 bits of the 76th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8009DC
i_emac_emac1	0xFF802000	0xFF8029DC
i_emac_emac2	0xFF804000	0xFF8049DC

Offset: 0x9DC

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address75_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address75 [31:0] This field contains the lower 32 bits of the 76th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address76_high

Register 632 (MAC Address76 High Register)

The MAC Address76 High register holds the upper 16 bits of the 77th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address76 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address76 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8009E0
i_emac_emac1	0xFF802000	0xFF8029E0
i_emac_emac2	0xFF804000	0xFF8049E0

Offset: 0x9E0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address76_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 77th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. Value: Description 0x0: DISABLED 0x1: ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address76 [47:32] This field contains the upper 16 bits (47:32) of the 77th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address76_low

Register 633 (MAC Address76 Low Register)

The MAC Address76 Low register holds the lower 32 bits of the 77th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8009E4
i_emac_emac1	0xFF802000	0xFF8029E4
i_emac_emac2	0xFF804000	0xFF8049E4

Offset: 0x9E4

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address76_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address76 [31:0] This field contains the lower 32 bits of the 77th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address77_high

Register 634 (MAC Address77 High Register)

The MAC Address77 High register holds the upper 16 bits of the 78th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address77 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address77 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8009E8
i_emac_emac1	0xFF802000	0xFF8029E8
i_emac_emac2	0xFF804000	0xFF8049E8

Offset: 0x9E8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address77_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 78th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address77 [47:32] This field contains the upper 16 bits (47:32) of the 78th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address77_low

Register 635 (MAC Address77 Low Register)

The MAC Address77 Low register holds the lower 32 bits of the 78th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8009EC
i_emac_emac1	0xFF802000	0xFF8029EC
i_emac_emac2	0xFF804000	0xFF8049EC

Offset: 0x9EC

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address77_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address77 [31:0] This field contains the lower 32 bits of the 78th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address78_high

Register 636 (MAC Address78 High Register)

The MAC Address78 High register holds the upper 16 bits of the 79th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address78 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address78 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8009F0
i_emac_emac1	0xFF802000	0xFF8029F0
i_emac_emac2	0xFF804000	0xFF8049F0

Offset: 0x9F0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address78_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 79th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address78 [47:32] This field contains the upper 16 bits (47:32) of the 79th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address78_low

Register 637 (MAC Address78 Low Register)

The MAC Address78 Low register holds the lower 32 bits of the 79th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8009F4
i_emac_emac1	0xFF802000	0xFF8029F4
i_emac_emac2	0xFF804000	0xFF8049F4

Offset: 0x9F4

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address78_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address78 [31:0] This field contains the lower 32 bits of the 79th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address79_high

Register 638 (MAC Address79 High Register)

The MAC Address79 High register holds the upper 16 bits of the 80th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address79 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address79 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8009F8
i_emac_emac1	0xFF802000	0xFF8029F8
i_emac_emac2	0xFF804000	0xFF8049F8

Offset: 0x9F8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address79_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 80th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. Value: Description 0x0: DISABLED 0x1: ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address79 [47:32] This field contains the upper 16 bits (47:32) of the 80th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address79_low

Register 639 (MAC Address79 Low Register)

The MAC Address79 Low register holds the lower 32 bits of the 80th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF8009FC
i_emac_emac1	0xFF802000	0xFF8029FC
i_emac_emac2	0xFF804000	0xFF8049FC

Offset: 0x9FC

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address79_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address79 [31:0] This field contains the lower 32 bits of the 80th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address80_high

Register 640 (MAC Address80 High Register)

The MAC Address80 High register holds the upper 16 bits of the 81st 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address80 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address80 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800A00
i_emac_emac1	0xFF802000	0xFF802A00
i_emac_emac2	0xFF804000	0xFF804A00

Offset: 0xA00

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address80_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 81st MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address80 [47:32] This field contains the upper 16 bits (47:32) of the 81st 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address80_low

Register 641 (MAC Address80 Low Register)

The MAC Address80 Low register holds the lower 32 bits of the 81st 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800A04
i_emac_emac1	0xFF802000	0xFF802A04
i_emac_emac2	0xFF804000	0xFF804A04

Offset: 0xA04

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address80_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address80 [31:0] This field contains the lower 32 bits of the 81st 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address81_high

Register 642 (MAC Address81 High Register)

The MAC Address81 High register holds the upper 16 bits of the 82nd 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address81 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address81 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800A08
i_emac_emac1	0xFF802000	0xFF802A08
i_emac_emac2	0xFF804000	0xFF804A08

Offset: 0xA08

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address81_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 82nd MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address81 [47:32] This field contains the upper 16 bits (47:32) of the 82nd 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address81_low

Register 643 (MAC Address81 Low Register)

The MAC Address81 Low register holds the lower 32 bits of the 82nd 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800A0C
i_emac_emac1	0xFF802000	0xFF802A0C
i_emac_emac2	0xFF804000	0xFF804A0C

Offset: 0xA0C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address81_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address81 [31:0] This field contains the lower 32 bits of the 82nd 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address82_high

Register 644 (MAC Address82 High Register)

The MAC Address82 High register holds the upper 16 bits of the 83rd 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address82 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address82 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800A10
i_emac_emac1	0xFF802000	0xFF802A10
i_emac_emac2	0xFF804000	0xFF804A10

Offset: 0xA10

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address82_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address82 [47:32] This field contains the upper 16 bits (47:32) of the 83rd 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address82_low

Register 645 (MAC Address82 Low Register)

The MAC Address82 Low register holds the lower 32 bits of the 83rd 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800A14
i_emac_emac1	0xFF802000	0xFF802A14
i_emac_emac2	0xFF804000	0xFF804A14

Offset: 0xA14

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address82_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address82 [31:0] This field contains the lower 32 bits of the 83rd 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address83_high

Register 646 (MAC Address83 High Register)

The MAC Address83 High register holds the upper 16 bits of the 84th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address83 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address83 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800A18
i_emac_emac1	0xFF802000	0xFF802A18
i_emac_emac2	0xFF804000	0xFF804A18

Offset: 0xA18

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address83_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 84th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address83 [47:32] This field contains the upper 16 bits (47:32) of the 84th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address83_low

Register 647 (MAC Address83 Low Register)

The MAC Address83 Low register holds the lower 32 bits of the 84th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800A1C
i_emac_emac1	0xFF802000	0xFF802A1C
i_emac_emac2	0xFF804000	0xFF804A1C

Offset: 0xA1C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address83_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address83 [31:0] This field contains the lower 32 bits of the 84th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address84_high

Register 648 (MAC Address84 High Register)

The MAC Address84 High register holds the upper 16 bits of the 85th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address84 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address84 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800A20
i_emac_emac1	0xFF802000	0xFF802A20
i_emac_emac2	0xFF804000	0xFF804A20

Offset: 0xA20

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address84_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 85th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address84 [47:32] This field contains the upper 16 bits (47:32) of the 85th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address84_low

Register 649 (MAC Address84 Low Register)

The MAC Address84 Low register holds the lower 32 bits of the 85th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800A24
i_emac_emac1	0xFF802000	0xFF802A24
i_emac_emac2	0xFF804000	0xFF804A24

Offset: 0xA24

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address84_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address84 [31:0] This field contains the lower 32 bits of the 85th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address85_high

Register 650 (MAC Address85 High Register)

The MAC Address85 High register holds the upper 16 bits of the 86th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address85 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address85 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800A28
i_emac_emac1	0xFF802000	0xFF802A28
i_emac_emac2	0xFF804000	0xFF804A28

Offset: 0xA28

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address85_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 86th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. Value: Description 0x0: DISABLED 0x1: ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address85 [47:32] This field contains the upper 16 bits (47:32) of the 86th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address85_low

Register 651 (MAC Address85 Low Register)

The MAC Address85 Low register holds the lower 32 bits of the 86th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800A2C
i_emac_emac1	0xFF802000	0xFF802A2C
i_emac_emac2	0xFF804000	0xFF804A2C

Offset: 0xA2C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address85_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address85 [31:0] This field contains the lower 32 bits of the 86th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address86_high

Register 652 (MAC Address86 High Register)

The MAC Address86 High register holds the upper 16 bits of the 87th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address86 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address86 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800A30
i_emac_emac1	0xFF802000	0xFF802A30
i_emac_emac2	0xFF804000	0xFF804A30

Offset: 0xA30

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address86_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 87th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address86 [47:32] This field contains the upper 16 bits (47:32) of the 87th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address86_low

Register 653 (MAC Address86 Low Register)

The MAC Address86 Low register holds the lower 32 bits of the 87th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800A34
i_emac_emac1	0xFF802000	0xFF802A34
i_emac_emac2	0xFF804000	0xFF804A34

Offset: 0xA34

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address86_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address86 [31:0] This field contains the lower 32 bits of the 87th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address87_high

Register 654 (MAC Address87 High Register)

The MAC Address87 High register holds the upper 16 bits of the 88th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address87 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address87 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800A38
i_emac_emac1	0xFF802000	0xFF802A38
i_emac_emac2	0xFF804000	0xFF804A38

Offset: 0xA38

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address87_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 88th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address87 [47:32] This field contains the upper 16 bits (47:32) of the 88th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address87_low

Register 655 (MAC Address87 Low Register)

The MAC Address87 Low register holds the lower 32 bits of the 88th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800A3C
i_emac_emac1	0xFF802000	0xFF802A3C
i_emac_emac2	0xFF804000	0xFF804A3C

Offset: 0xA3C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address87_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address87 [31:0] This field contains the lower 32 bits of the 88th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address88_high

Register 656 (MAC Address88 High Register)

The MAC Address88 High register holds the upper 16 bits of the 89th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address88 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address88 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800A40
i_emac_emac1	0xFF802000	0xFF802A40
i_emac_emac2	0xFF804000	0xFF804A40

Offset: 0xA40

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address88_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 89th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address88 [47:32] This field contains the upper 16 bits (47:32) of the 89th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address88_low

Register 657 (MAC Address88 Low Register)

The MAC Address88 Low register holds the lower 32 bits of the 89th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800A44
i_emac_emac1	0xFF802000	0xFF802A44
i_emac_emac2	0xFF804000	0xFF804A44

Offset: 0xA44

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address88_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address88 [31:0] This field contains the lower 32 bits of the 89th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address89_high

Register 658 (MAC Address89 High Register)

The MAC Address89 High register holds the upper 16 bits of the 90th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address89 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address89 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800A48
i_emac_emac1	0xFF802000	0xFF802A48
i_emac_emac2	0xFF804000	0xFF804A48

Offset: 0xA48

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address89_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 90th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address89 [47:32] This field contains the upper 16 bits (47:32) of the 90th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address89_low

Register 659 (MAC Address89 Low Register)

The MAC Address89 Low register holds the lower 32 bits of the 90th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800A4C
i_emac_emac1	0xFF802000	0xFF802A4C
i_emac_emac2	0xFF804000	0xFF804A4C

Offset: 0xA4C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address89_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address89 [31:0] This field contains the lower 32 bits of the 90th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address90_high

Register 660 (MAC Address90 High Register)

The MAC Address90 High register holds the upper 16 bits of the 91st 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address90 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address90 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800A50
i_emac_emac1	0xFF802000	0xFF802A50
i_emac_emac2	0xFF804000	0xFF804A50

Offset: 0xA50

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address90_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 91st MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address90 [47:32] This field contains the upper 16 bits (47:32) of the 91st 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address90_low

Register 661 (MAC Address90 Low Register)

The MAC Address90 Low register holds the lower 32 bits of the 91st 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800A54
i_emac_emac1	0xFF802000	0xFF802A54
i_emac_emac2	0xFF804000	0xFF804A54

Offset: 0xA54

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address90_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address90 [31:0] This field contains the lower 32 bits of the 91st 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address91_high

Register 662 (MAC Address91 High Register)

The MAC Address91 High register holds the upper 16 bits of the 92nd 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address32 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address91 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800A58
i_emac_emac1	0xFF802000	0xFF802A58
i_emac_emac2	0xFF804000	0xFF804A58

Offset: 0xA58

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address91_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 92nd MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address91 [47:32] This field contains the upper 16 bits (47:32) of the 92nd 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address91_low

Register 663 (MAC Address91 Low Register)

The MAC Address91 Low register holds the lower 32 bits of the 92nd 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800A5C
i_emac_emac1	0xFF802000	0xFF802A5C
i_emac_emac2	0xFF804000	0xFF804A5C

Offset: 0xA5C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address91_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address91 [31:0] This field contains the lower 32 bits of the 92nd 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address92_high

Register 664 (MAC Address92 High Register)

The MAC Address92 High register holds the upper 16 bits of the 93rd 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address92 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address92 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800A60
i_emac_emac1	0xFF802000	0xFF802A60
i_emac_emac2	0xFF804000	0xFF804A60

Offset: 0xA60

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address92_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 93rd MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address92 [47:32] This field contains the upper 16 bits (47:32) of the 93rd 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address92_low

Register 665 (MAC Address92 Low Register)

The MAC Address92 Low register holds the lower 32 bits of the 93rd 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800A64
i_emac_emac1	0xFF802000	0xFF802A64
i_emac_emac2	0xFF804000	0xFF804A64

Offset: 0xA64

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address92_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address92 [31:0] This field contains the lower 32 bits of the 93rd 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address93_high

Register 666 (MAC Address93 High Register)

The MAC Address93 High register holds the upper 16 bits of the 94th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address93 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address93 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800A68
i_emac_emac1	0xFF802000	0xFF802A68
i_emac_emac2	0xFF804000	0xFF804A68

Offset: 0xA68

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address93_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 94th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address93 [47:32] This field contains the upper 16 bits (47:32) of the 94th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address93_low

Register 667 (MAC Address93 Low Register)

The MAC Address93 Low register holds the lower 32 bits of the 94th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800A6C
i_emac_emac1	0xFF802000	0xFF802A6C
i_emac_emac2	0xFF804000	0xFF804A6C

Offset: 0xA6C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address93_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address93 [31:0] This field contains the lower 32 bits of the 94th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address94_high

Register 668 (MAC Address94 High Register)

The MAC Address94 High register holds the upper 16 bits of the 95th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address94 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address94 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800A70
i_emac_emac1	0xFF802000	0xFF802A70
i_emac_emac2	0xFF804000	0xFF804A70

Offset: 0xA70

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address94_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 95th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address94 [47:32] This field contains the upper 16 bits (47:32) of the 95th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address94_low

Register 669 (MAC Address94 Low Register)

The MAC Address94 Low register holds the lower 32 bits of the 95th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800A74
i_emac_emac1	0xFF802000	0xFF802A74
i_emac_emac2	0xFF804000	0xFF804A74

Offset: 0xA74

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address94_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address94 [31:0] This field contains the lower 32 bits of the 95th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address95_high

Register 670 (MAC Address95 High Register)

The MAC Address95 High register holds the upper 16 bits of the 96th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address95 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address95 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800A78
i_emac_emac1	0xFF802000	0xFF802A78
i_emac_emac2	0xFF804000	0xFF804A78

Offset: 0xA78

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address95_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 96th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address95 [47:32] This field contains the upper 16 bits (47:32) of the 96th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address95_low

Register 671 (MAC Address95 Low Register)

The MAC Address95 Low register holds the lower 32 bits of the 96th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800A7C
i_emac_emac1	0xFF802000	0xFF802A7C
i_emac_emac2	0xFF804000	0xFF804A7C

Offset: 0xA7C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address95_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address95 [31:0] This field contains the lower 32 bits of the 96th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address96_high

Register 672 (MAC Address96 High Register)

The MAC Address96 High register holds the upper 16 bits of the 97th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address96 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address96 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800A80
i_emac_emac1	0xFF802000	0xFF802A80
i_emac_emac2	0xFF804000	0xFF804A80

Offset: 0xA80

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address96_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 97th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address96 [47:32] This field contains the upper 16 bits (47:32) of the 97th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address96_low

Register 673 (MAC Address96 Low Register)

The MAC Address96 Low register holds the lower 32 bits of the 97th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800A84
i_emac_emac1	0xFF802000	0xFF802A84
i_emac_emac2	0xFF804000	0xFF804A84

Offset: 0xA84

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address96_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address96 [31:0] This field contains the lower 32 bits of the 97th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address97_high

Register 674 (MAC Address97 High Register)

The MAC Address97 High register holds the upper 16 bits of the 98th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address97 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address97 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800A88
i_emac_emac1	0xFF802000	0xFF802A88
i_emac_emac2	0xFF804000	0xFF804A88

Offset: 0xA88

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address97_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 98th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address97 [47:32] This field contains the upper 16 bits (47:32) of the 98th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address97_low

Register 675 (MAC Address97 Low Register)

The MAC Address97 Low register holds the lower 32 bits of the 98th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800A8C
i_emac_emac1	0xFF802000	0xFF802A8C
i_emac_emac2	0xFF804000	0xFF804A8C

Offset: 0xA8C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address97_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address97 [31:0] This field contains the lower 32 bits of the 98th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address98_high

Register 676 (MAC Address98 High Register)

The MAC Address99 High register holds the upper 16 bits of the 100th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address99 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address99 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800A90
i_emac_emac1	0xFF802000	0xFF802A90
i_emac_emac2	0xFF804000	0xFF804A90

Offset: 0xA90

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address98_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 99th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address98 [47:32] This field contains the upper 16 bits (47:32) of the 99th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address98_low

Register 677 (MAC Address98 Low Register)

The MAC Address98 Low register holds the lower 32 bits of the 99th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800A94
i_emac_emac1	0xFF802000	0xFF802A94
i_emac_emac2	0xFF804000	0xFF804A94

Offset: 0xA94

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address98_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address98 [31:0] This field contains the lower 32 bits of the 99th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address99_high

Register 678 (MAC Address99 High Register)

The MAC Address99 High register holds the upper 16 bits of the 6-byte 100th MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address99 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address99 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800A98
i_emac_emac1	0xFF802000	0xFF802A98
i_emac_emac2	0xFF804000	0xFF804A98

Offset: 0xA98

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address99_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 100th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address99 [47:32] This field contains the upper 16 bits (47:32) of the 100th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address99_low

Register 679 (MAC Address99 Low Register)

The MAC Address99 Low register holds the lower 32 bits of the 100th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800A9C
i_emac_emac1	0xFF802000	0xFF802A9C
i_emac_emac2	0xFF804000	0xFF804A9C

Offset: 0xA9C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address99_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address99 [31:0] This field contains the lower 32 bits of the 100th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address100_high

Register 680 (MAC Address100 High Register)

The MAC Address100 High register holds the upper 16 bits of the 101th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address100 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address100 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800AA0
i_emac_emac1	0xFF802000	0xFF802AA0
i_emac_emac2	0xFF804000	0xFF804AA0

Offset: 0xAA0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address100_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 101th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address100 [47:32] This field contains the upper 16 bits (47:32) of the 101th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address100_low

Register 681 (MAC Address100 Low Register)

The MAC Address100 Low register holds the lower 32 bits of the 101th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800AA4
i_emac_emac1	0xFF802000	0xFF802AA4
i_emac_emac2	0xFF804000	0xFF804AA4

Offset: 0xAA4

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address100_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address100 [31:0] This field contains the lower 32 bits of the 101th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address101_high

Register 682 (MAC Address101 High Register)

The MAC Address101 High register holds the upper 16 bits of the 102nd 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address101 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address101 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800AA8
i_emac_emac1	0xFF802000	0xFF802AA8
i_emac_emac2	0xFF804000	0xFF804AA8

Offset: 0xAA8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address101_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 102nd MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address101 [47:32] This field contains the upper 16 bits (47:32) of the 102nd 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address101_low

Register 683 (MAC Address101 Low Register)

The MAC Address101 Low register holds the lower 32 bits of the 102nd 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800AAC
i_emac_emac1	0xFF802000	0xFF802AAC
i_emac_emac2	0xFF804000	0xFF804AAC

Offset: 0xAAC

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address101_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address101 [31:0] This field contains the lower 32 bits of the 102nd 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address102_high

Register 684 (MAC Address102 High Register)

The MAC Address102 High register holds the upper 16 bits of the 6-byte 103rd MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address102 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address102 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800AB0
i_emac_emac1	0xFF802000	0xFF802AB0
i_emac_emac2	0xFF804000	0xFF804AB0

Offset: 0xAB0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address102_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 103rd MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address102 [47:32] This field contains the upper 16 bits (47:32) of the 103rd 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address102_low

Register 685 (MAC Address102 Low Register)

The MAC Address102 Low register holds the lower 32 bits of the 103rd 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800AB4
i_emac_emac1	0xFF802000	0xFF802AB4
i_emac_emac2	0xFF804000	0xFF804AB4

Offset: 0xAB4

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address102_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address102 [31:0] This field contains the lower 32 bits of the 103rd 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address103_high

Register 686 (MAC Address103 High Register)

The MAC Address103 High register holds the upper 16 bits of the 6-byte 104th MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address103 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address103 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800AB8
i_emac_emac1	0xFF802000	0xFF802AB8
i_emac_emac2	0xFF804000	0xFF804AB8

Offset: 0xAB8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address103_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 104th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address103 [47:32] This field contains the upper 16 bits (47:32) of the 104th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address103_low

Register 687 (MAC Address103 Low Register)

The MAC Address103 Low register holds the lower 32 bits of the 104th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800ABC
i_emac_emac1	0xFF802000	0xFF802ABC
i_emac_emac2	0xFF804000	0xFF804ABC

Offset: 0xABC

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address103_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address103 [31:0] This field contains the lower 32 bits of the 104th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address104_high

Register 688 (MAC Address104 High Register)

The MAC Address104 High register holds the upper 16 bits of the 105th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address104 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address104 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800AC0
i_emac_emac1	0xFF802000	0xFF802AC0
i_emac_emac2	0xFF804000	0xFF804AC0

Offset: 0xAC0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address104_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 105th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address104 [47:32] This field contains the upper 16 bits (47:32) of the 105th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address104_low

Register 689 (MAC Address104 Low Register)

The MAC Address104 Low register holds the lower 32 bits of the 105th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800AC4
i_emac_emac1	0xFF802000	0xFF802AC4
i_emac_emac2	0xFF804000	0xFF804AC4

Offset: 0xAC4

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address104_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address104 [31:0] This field contains the lower 32 bits of the 105th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address105_high

Register 690 (MAC Address105 High Register)

The MAC Address105 High register holds the upper 16 bits of the 106th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address105 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address105 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800AC8
i_emac_emac1	0xFF802000	0xFF802AC8
i_emac_emac2	0xFF804000	0xFF804AC8

Offset: 0xAC8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address105_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 106th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address105 [47:32] This field contains the upper 16 bits (47:32) of the 106th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address105_low

Register 691 (MAC Address105 Low Register)

The MAC Address105 Low register holds the lower 32 bits of the 106th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800ACC
i_emac_emac1	0xFF802000	0xFF802ACC
i_emac_emac2	0xFF804000	0xFF804ACC

Offset: 0xACC

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address105_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address105 [31:0] This field contains the lower 32 bits of the 106th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address106_high

Register 692 (MAC Address106 High Register)

The MAC Address106 High register holds the upper 16 bits of the 107th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address106 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address106 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800AD0
i_emac_emac1	0xFF802000	0xFF802AD0
i_emac_emac2	0xFF804000	0xFF804AD0

Offset: 0xAD0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address106_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 107th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address106 [47:32] This field contains the upper 16 bits (47:32) of the 107th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address106_low

Register 693 (MAC Address106 Low Register)

The MAC Address106 Low register holds the lower 32 bits of the 107th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800AD4
i_emac_emac1	0xFF802000	0xFF802AD4
i_emac_emac2	0xFF804000	0xFF804AD4

Offset: 0xAD4

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address106_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address106 [31:0] This field contains the lower 32 bits of the 107th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address107_high

Register 694 (MAC Address107 High Register)

The MAC Address107 High register holds the upper 16 bits of the 108th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address107 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address107 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800AD8
i_emac_emac1	0xFF802000	0xFF802AD8
i_emac_emac2	0xFF804000	0xFF804AD8

Offset: 0xAD8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address107_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 108th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address107 [47:32] This field contains the upper 16 bits (47:32) of the 108th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address107_low

Register 695 (MAC Address107 Low Register)

The MAC Address107 Low register holds the lower 32 bits of the 108th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800ADC
i_emac_emac1	0xFF802000	0xFF802ADC
i_emac_emac2	0xFF804000	0xFF804ADC

Offset: 0xADC

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address107_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address107 [31:0] This field contains the lower 32 bits of the 108th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address108_high

Register 696 (MAC Address108 High Register)

The MAC Address108 High register holds the upper 16 bits of the 109th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address108 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address108 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800AE0
i_emac_emac1	0xFF802000	0xFF802AE0
i_emac_emac2	0xFF804000	0xFF804AE0

Offset: 0xAE0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address108_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 109th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address108 [47:32] This field contains the upper 16 bits (47:32) of the 109th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address108_low

Register 697 (MAC Address108 Low Register)

The MAC Address108 Low register holds the lower 32 bits of the 109th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800AE4
i_emac_emac1	0xFF802000	0xFF802AE4
i_emac_emac2	0xFF804000	0xFF804AE4

Offset: 0xAE4

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address108_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address108 [31:0] This field contains the lower 32 bits of the 109th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address109_high

Register 698 (MAC Address109 High Register)

The MAC Address109 High register holds the upper 16 bits of the 110th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address109 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address109 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800AE8
i_emac_emac1	0xFF802000	0xFF802AE8
i_emac_emac2	0xFF804000	0xFF804AE8

Offset: 0xAE8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address109_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 110th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address109 [47:32] This field contains the upper 16 bits (47:32) of the 110th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address109_low

Register 699 (MAC Address109 Low Register)

The MAC Address109 Low register holds the lower 32 bits of the 110th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800AEC
i_emac_emac1	0xFF802000	0xFF802AEC
i_emac_emac2	0xFF804000	0xFF804AEC

Offset: 0xAEC

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address109_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address109 [31:0] This field contains the lower 32 bits of the 110th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address110_high

Register XXX (MAC AddressXX High Register)

The MAC Address110 High register holds the upper 16 bits of the 111th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address110 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address110 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800AF0
i_emac_emac1	0xFF802000	0xFF802AF0
i_emac_emac2	0xFF804000	0xFF804AF0

Offset: 0xAF0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address110_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 111th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address110 [47:32] This field contains the upper 16 bits (47:32) of the 111th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address110_low

Register 700 (MAC Address110 Low Register)

The MAC Address110 Low register holds the lower 32 bits of the 111th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800AF4
i_emac_emac1	0xFF802000	0xFF802AF4
i_emac_emac2	0xFF804000	0xFF804AF4

Offset: 0xAF4

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address110_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address110 [31:0] This field contains the lower 32 bits of the 111th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address111_high

Register 701 (MAC Address111 High Register)

The MAC Address111 High register holds the upper 16 bits of the 6-byte 112th MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address111 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address111 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800AF8
i_emac_emac1	0xFF802000	0xFF802AF8
i_emac_emac2	0xFF804000	0xFF804AF8

Offset: 0xAF8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address111_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 112th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address111 [47:32] This field contains the upper 16 bits (47:32) of the 112th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address111_low

Register 702 (MAC Address111 Low Register)

The MAC Address111 Low register holds the lower 32 bits of the 112th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800AFC
i_emac_emac1	0xFF802000	0xFF802AFC
i_emac_emac2	0xFF804000	0xFF804AFC

Offset: 0xAFC

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address111_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address111 [31:0] This field contains the lower 32 bits of the 112th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address112_high

Register 703 (MAC Address112 High Register)

The MAC Address112 High register holds the upper 16 bits of the 113th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address112 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address112 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800B00
i_emac_emac1	0xFF802000	0xFF802B00
i_emac_emac2	0xFF804000	0xFF804B00

Offset: 0xB00

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address112_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 113th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address112 [47:32] This field contains the upper 16 bits (47:32) of the 113th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address112_low

Register 704 (MAC Address112 Low Register)

The MAC Address112 Low register holds the lower 32 bits of the 113th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800B04
i_emac_emac1	0xFF802000	0xFF802B04
i_emac_emac2	0xFF804000	0xFF804B04

Offset: 0xB04

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address112_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address112 [31:0] This field contains the lower 32 bits of the 113th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address113_high

Register 705 (MAC Address113 High Register)

The MAC Address113 High register holds the upper 16 bits of the 114th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address113 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address113 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800B08
i_emac_emac1	0xFF802000	0xFF802B08
i_emac_emac2	0xFF804000	0xFF804B08

Offset: 0xB08

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address113_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 114th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address113 [47:32] This field contains the upper 16 bits (47:32) of the 114th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address113_low

Register 706 (MAC Address113 Low Register)

The MAC Address113 Low register holds the lower 32 bits of the 114th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800B0C
i_emac_emac1	0xFF802000	0xFF802B0C
i_emac_emac2	0xFF804000	0xFF804B0C

Offset: 0xB0C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address113_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address113 [31:0] This field contains the lower 32 bits of the 114th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address114_high

Register 707 (MAC Address114 High Register)

The MAC Address114 High register holds the upper 16 bits of the 115th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address114 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address114 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800B10
i_emac_emac1	0xFF802000	0xFF802B10
i_emac_emac2	0xFF804000	0xFF804B10

Offset: 0xB10

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address114_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 115th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address114 [47:32] This field contains the upper 16 bits (47:32) of the 115th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address114_low

Register 708 (MAC Address114 Low Register)

The MAC Address114 Low register holds the lower 32 bits of the 115th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800B14
i_emac_emac1	0xFF802000	0xFF802B14
i_emac_emac2	0xFF804000	0xFF804B14

Offset: 0xB14

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address114_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address114 [31:0] This field contains the lower 32 bits of the 115th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address115_high

Register 709 (MAC Address115 High Register)

The MAC Address115 High register holds the upper 16 bits of the 116th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address115 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address115 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800B18
i_emac_emac1	0xFF802000	0xFF802B18
i_emac_emac2	0xFF804000	0xFF804B18

Offset: 0xB18

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address115_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 116th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address115 [47:32] This field contains the upper 16 bits (47:32) of the 116th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address115_low

Register 710 (MAC Address115 Low Register)

The MAC Address115 Low register holds the lower 32 bits of the 116th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800B1C
i_emac_emac1	0xFF802000	0xFF802B1C
i_emac_emac2	0xFF804000	0xFF804B1C

Offset: 0xB1C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address115_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address115 [31:0] This field contains the lower 32 bits of the 116th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address116_high

Register 711 (MAC Address116 High Register)

The MAC Address116 High register holds the upper 16 bits of the 117th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address116 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address116 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800B20
i_emac_emac1	0xFF802000	0xFF802B20
i_emac_emac2	0xFF804000	0xFF804B20

Offset: 0xB20

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address116_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 117th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address116 [47:32] This field contains the upper 16 bits (47:32) of the 117th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address116_low

Register 712 (MAC Address116 Low Register)

The MAC Address116 Low register holds the lower 32 bits of the 117th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800B24
i_emac_emac1	0xFF802000	0xFF802B24
i_emac_emac2	0xFF804000	0xFF804B24

Offset: 0xB24

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address116_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address116 [31:0] This field contains the lower 32 bits of the 117th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address117_high

Register 713 (MAC Address117 High Register)

The MAC Address117 High register holds the upper 16 bits of the 118th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode of the MAC Address117 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address117 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800B28
i_emac_emac1	0xFF802000	0xFF802B28
i_emac_emac2	0xFF804000	0xFF804B28

Offset: 0xB28

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address117_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 118th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address117 [47:32] This field contains the upper 16 bits (47:32) of the 118th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address117_low

Register 714 (MAC Address117 Low Register)

The MAC Address117 Low register holds the lower 32 bits of the 118th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800B2C
i_emac_emac1	0xFF802000	0xFF802B2C
i_emac_emac2	0xFF804000	0xFF804B2C

Offset: 0xB2C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address117_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address117 [31:0] This field contains the lower 32 bits of the 118t118thh 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address118_high

Register 715 (MAC Address118 High Register)

The MAC Address118 High register holds the upper 16 bits of the 119th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address118 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address118 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800B30
i_emac_emac1	0xFF802000	0xFF802B30
i_emac_emac2	0xFF804000	0xFF804B30

Offset: 0xB30

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address118_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 119th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address118 [47:32] This field contains the upper 16 bits (47:32) of the 119th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address118_low

Register 716 (MAC Address118 Low Register)

The MAC Address118 Low register holds the lower 32 bits of the 119th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800B34
i_emac_emac1	0xFF802000	0xFF802B34
i_emac_emac2	0xFF804000	0xFF804B34

Offset: 0xB34

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address118_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address118 [31:0] This field contains the lower 32 bits of the 119th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address119_high

Register 717 (MAC Address119 High Register)

The MAC Address119 High register holds the upper 16 bits of the 120th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address119 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address119 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800B38
i_emac_emac1	0xFF802000	0xFF802B38
i_emac_emac2	0xFF804000	0xFF804B38

Offset: 0xB38

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address119_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 120th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address119 [47:32] This field contains the upper 16 bits (47:32) of the 120th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address119_low

Register 718 (MAC Address119 Low Register)

The MAC Address119 Low register holds the lower 32 bits of the 120th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800B3C
i_emac_emac1	0xFF802000	0xFF802B3C
i_emac_emac2	0xFF804000	0xFF804B3C

Offset: 0xB3C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address119_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address119 [31:0] This field contains the lower 32 bits of the 120th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address120_high

Register 719 (MAC Address120 High Register)

The MAC Address120 High register holds the upper 16 bits of the 6-byte 121st MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address120 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address120 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800B40
i_emac_emac1	0xFF802000	0xFF802B40
i_emac_emac2	0xFF804000	0xFF804B40

Offset: 0xB40

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address120_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 121st MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address120 [47:32] This field contains the upper 16 bits (47:32) of the 6-byte 121st MAC address.	RW	0xFFFF

gmacgrp_mac_address120_low

Register 720 (MAC Address120 Low Register)

The MAC Address120 Low register holds the lower 32 bits of the 121st 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800B44
i_emac_emac1	0xFF802000	0xFF802B44
i_emac_emac2	0xFF804000	0xFF804B44

Offset: 0xB44

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address120_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address120 [31:0] This field contains the lower 32 bits of the 122nd 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address121_high

Register 721 (MAC Address121 High Register)

The MAC Address121 High register holds the upper 16 bits of the 122nd 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address121 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address121 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800B48
i_emac_emac1	0xFF802000	0xFF802B48
i_emac_emac2	0xFF804000	0xFF804B48

Offset: 0xB48

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address121_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 122nd MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. Value: Description 0x0: DISABLED 0x1: ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address121 [47:32] This field contains the upper 16 bits (47:32) of the 122nd 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address121_low

Register 722 (MAC Address121 Low Register)

The MAC Address121 Low register holds the lower 32 bits of the 122nd 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800B4C
i_emac_emac1	0xFF802000	0xFF802B4C
i_emac_emac2	0xFF804000	0xFF804B4C

Offset: 0xB4C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address121_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address121 [31:0] This field contains the lower 32 bits of the 122nd 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address122_high

Register 723 (MAC Address122 High Register)

The MAC Address122 High register holds the upper 16 bits of the 123rd 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address122 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address122 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800B50
i_emac_emac1	0xFF802000	0xFF802B50
i_emac_emac2	0xFF804000	0xFF804B50

Offset: 0xB50

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address122_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 123rd MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address122 [47:32] This field contains the upper 16 bits (47:32) of the 123rd 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address122_low

Register 724 (MAC Address122 Low Register)

The MAC Address122 Low register holds the lower 32 bits of the 123rd 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800B54
i_emac_emac1	0xFF802000	0xFF802B54
i_emac_emac2	0xFF804000	0xFF804B54

Offset: 0xB54

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address122_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address122 [31:0] This field contains the lower 32 bits of the 123rd 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address123_high

Register 725 (MAC Address123 High Register)

The MAC Address123 High register holds the upper 16 bits of the 124th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address123 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address123 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800B58
i_emac_emac1	0xFF802000	0xFF802B58
i_emac_emac2	0xFF804000	0xFF804B58

Offset: 0xB58

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address123_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 124th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address123 [47:32] This field contains the upper 16 bits (47:32) of the 124th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address123_low

Register 726 (MAC AddressXX 123 Register)

The MAC Address123 Low register holds the lower 32 bits of the 124th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800B5C
i_emac_emac1	0xFF802000	0xFF802B5C
i_emac_emac2	0xFF804000	0xFF804B5C

Offset: 0xB5C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address123_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address123 [31:0] This field contains the lower 32 bits of the 124th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address124_high

Register 727 (MAC Address124 High Register)

The MAC Address124 High register holds the upper 16 bits of the 125th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address124 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address124 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800B60
i_emac_emac1	0xFF802000	0xFF802B60
i_emac_emac2	0xFF804000	0xFF804B60

Offset: 0xB60

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address124_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 125th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address124 [47:32] This field contains the upper 16 bits (47:32) of the 125th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address124_low

Register 728 (MAC Address124 Low Register)

The MAC Address124 Low register holds the lower 32 bits of the 125th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800B64
i_emac_emac1	0xFF802000	0xFF802B64
i_emac_emac2	0xFF804000	0xFF804B64

Offset: 0xB64

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address124_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address124 [31:0] This field contains the lower 32 bits of the 125th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address125_high

Register 729 (MAC Address125 High Register)

The MAC Address125 High register holds the upper 16 bits of the 126th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address125 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address125 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800B68
i_emac_emac1	0xFF802000	0xFF802B68
i_emac_emac2	0xFF804000	0xFF804B68

Offset: 0xB68

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address125_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 126th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address125 [47:32] This field contains the upper 16 bits (47:32) of the 126th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address125_low

Register 730 (MAC Address125 Low Register)

The MAC Address125 Low register holds the lower 32 bits of the 126th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800B6C
i_emac_emac1	0xFF802000	0xFF802B6C
i_emac_emac2	0xFF804000	0xFF804B6C

Offset: 0xB6C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address125_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address125 [31:0] This field contains the lower 32 bits of the 126th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address126_high

Register 731 (MAC Address126 High Register)

The MAC Address126 High register holds the upper 16 bits of the 127th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then
the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address126 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address126 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800B70
i_emac_emac1	0xFF802000	0xFF802B70
i_emac_emac2	0xFF804000	0xFF804B70

Offset: 0xB70

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address126_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 127th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address126 [47:32] This field contains the upper 16 bits (47:32) of the 127th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address126_low

Register 732 (MAC Address126 Low Register)

The MAC Address126 Low register holds the lower 32 bits of the 127th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800B74
i_emac_emac1	0xFF802000	0xFF802B74
i_emac_emac2	0xFF804000	0xFF804B74

Offset: 0xB74

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address126_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address126 [31:0] This field contains the lower 32 bits of the 127th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

gmacgrp_mac_address127_high

Register 733 (MAC Address127 High Register)

The MAC Address127 High register holds the upper 16 bits of the 128th 6-byte MAC address of the station.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) of the MAC Address127 Low Register are written. For proper synchronization updates, consecutive writes to this MAC Address127 Low Register must be performed after at least four clock cycles in the destination clock domain.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800B78
i_emac_emac1	0xFF802000	0xFF802B78
i_emac_emac2	0xFF804000	0xFF804B78

Offset: 0xB78

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ae RW 0x0	reserved_30_16 RO 0x0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrhi RW 0xFFFF															

gmacgrp_mac_address127_high Fields

Bit	Name	Description	Access	Reset
31	ae	Address Enable When this bit is set, the address filter module uses the 128th MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. Value: Description 0x0: DISABLED 0x1: ENABLED	RW	0x0
30:16	reserved_30_16	Reserved	RO	0x0
15:0	addrhi	MAC Address127 [47:32] This field contains the upper 16 bits (47:32) of the 128th 6-byte MAC address.	RW	0xFFFF

gmacgrp_mac_address127_low

Register 734 (MAC Address127 Low Register)

The MAC Address127 Low register holds the lower 32 bits of the 128th 6-byte MAC address of the station.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF800B7C
i_emac_emac1	0xFF802000	0xFF802B7C
i_emac_emac2	0xFF804000	0xFF804B7C

Offset: 0xB7C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
addrlo RW 0xFFFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
addrlo RW 0xFFFFFFFF															

gmacgrp_mac_address127_low Fields

Bit	Name	Description	Access	Reset
31:0	addrlo	MAC Address127 [31:0] This field contains the lower 32 bits of the 128th 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process.	RW	0xFFFFFFFF

dmagrp_bus_mode

Register 0 (Bus Mode Register)

The Bus Mode register establishes the bus operating modes for the DMA.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF801000
i_emac_emac1	0xFF802000	0xFF803000
i_emac_emac2	0xFF804000	0xFF805000

Offset: 0x1000

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
reserved_31_26 RO 0x0						aal RW 0x0	eightxpbl RW 0x0	usp RW 0x0	rpbl RW 0x1						fb RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved_15_14 RO 0x0		pbl RW 0x1						atds RW 0x0	dsl RW 0x0					reserved_1 RO 0x0	swr RW 0x1

dmagrp_bus_mode Fields

Bit	Name	Description	Access	Reset
30:26	reserved_30_26	Reserved	RO	0x0
25	aal	Address Aligned Beats When this bit is set high and the FB bit is equal to 1, the AXI interface generates all bursts aligned to the start address LS bits. If the FB bit is equal to 0, the first burst (accessing the data buffer's start address) is not aligned, but subsequent bursts are aligned to the address. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
24	eightxpbl	PBLx8 Mode When set high, this bit multiplies the programmed PBL value (Bits[22:17] and Bits[13:8]) eight times. Therefore, the DMA transfers the data in 8, 16, 32, 64, 128, and 256 beats depending on the PBL value. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
23	usp	Use Seperate PBL When set high, this bit configures the Rx DMA to use the value configured in Bits[22:17] as PBL. The PBL value in Bits[13:8] is applicable only to the Tx DMA operations. When reset to low, the PBL value in Bits[13:8] is applicable for both DMA engines. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
22:17	rpbl	Rx DMA PBL This field indicates the maximum number of beats to be transferred in one Rx DMA transaction. This is the maximum value that is used in a single block Read or Write. The Rx DMA always attempts to burst as specified in the RPBL bit each time it starts a Burst transfer on the host bus. You can program RPBL with values of 1, 2, 4, 8, 16, and 32. Any other value results in undefined behavior. This field is valid and applicable only when USP is set high. **Value** **Description** 0x1 RXDMAPBL1 0x2 RXDMAPBL2 0x4 RXDMAPBL4 0x8 RXDMAPBL8 0x10 RXDMAPBL6 0x20 RXDMAPBL32	RW	0x1
16	fb	Fixed Burst This bit controls whether the AXI Master interface performs fixed burst transfers or not. When set, the interface uses only SINGLE, INCR4, INCR8, or INCR16 during start of the normal burst transfers. When reset, the AHB or AXI interface uses SINGLE and INCR burst transfer operations. For more information, see Bit 0 (UNDEF). **Value** **Description** 0x0 NONFB 0x1 FB1_4_8_16	RW	0x0
15:14	reserved_15_14	Reserved	RW	0x0

Bit	Name	Description	Access	Reset
13:8	pbl	Programmable Burst Length These bits indicate the maximum number of beats to be transferred in one DMA transaction. This is the maximum value that is used in a single block Read or Write. The DMA always attempts to burst as specified in PBL each time it starts a Burst transfer on the host bus. PBL can be programmed with permissible values of 1, 2, 4, 8, 16, and 32. Any other value results in undefined behavior. When USP is set high, this PBL value is applicable only for Tx DMA transactions. If the number of beats to be transferred is more than 32, then perform the following steps: **1.** Set the PBLx8 mode. **2.** Set the PBL. For example, if the maximum number of beats to be transferred is 64, then first set PBLx8 to 1 and then set PBL to 8. The PBL values have the following limitation: The maximum number of possible beats (PBL) is limited by the size of the Tx FIFO and Rx FIFO in the MTL layer and the data bus width on the DMA. The FIFO has a constraint that the maximum beat supported is half the depth of the FIFO, except when specified. For different data bus widths and FIFO sizes, the valid PBL range (including x8 mode) is provided in the following list. If the PBL is common for both transmit and receive DMA, the minimum Rx FIFO and Tx FIFO depths must be considered. **Note:** Note: In the half-duplex mode, the valid PBL range specified in the following list is applicable only for Tx FIFO.	RW	0x1

Bit	Name	Description	Access	Reset

		• 32-Bit Data Bus Width
		  • 128 Bytes FIFO Depth: In the full-duplex mode, the valid PBL range is 16 or less. In the half-duplex mode, the valid PBL range is 8 or less for the 10 or 100 Mbps mode.
		  • 256 Bytes FIFO Depth: In the full-duplex mode and the half-duplex (10 or 100 Mbps) modes, the valid PBL range is 32 or less.
		  • 512 Bytes FIFO Depth: In the full-duplex mode and the half-duplex (10 or 100 Mbps) modes, the valid PBL range is 64 or less.
		  • 1 KB FIFO Depth: In the full-duplex mode, the valid PBL range is 128 or less. In the half-duplex mode, the valid PBL range is 128 or less in the 10 or 100 Mbps mode and 64 or less in the 1000 Mbps mode.
		  • 2 KB and Higher FIFO Depth: All PBL values are supported in the full-duplex mode and half-duplex modes.		

Bit	Name	Description	Access	Reset
7	atds	Alternate Descriptor Size When set, the size of the alternate descriptor increases to 32 bytes (8 DWORDS). This is required when the Advanced Timestamp feature or the IPC Full Offload Engine (Type 2) is enabled in the receiver. The enhanced descriptor is not required if the Advanced Timestamp and IPC Full Checksum Offload (Type 2) features are not enabled. In such cases, you can use the 16 bytes descriptor to save 4 bytes of memory. When reset, the descriptor size reverts back to 4 DWORDs (16 bytes). This bit preserves the backward compatibility for the descriptor size. In versions prior to 3.50a, the descriptor size is 16 bytes for both normal and enhanced descriptor. In version 3.50a, descriptor size is increased to 32 bytes because of the Advanced Timestamp and IPC Full Checksum Offload Engine (Type 2) features. **Value** **Description** 0x0 CLEARRST 0x1 RESET	RW	0x0
6:2	dsl	Descriptor Skip Length This bit specifies the number of Words to skip between two unchained descriptors. The address skipping starts from the end of current descriptor to the start of next descriptor. When the DSL value is equal to zero, the descriptor table is taken as contiguous by the DMA in Ring mode.	RW	0x0
1	reserved_1	Reserved_1	RW	0x0

Bit	Name	Description	Access	Reset
0	swr	Software Reset When this bit is set, the MAC DMA Controller resets the logic and all internal registers of the MAC. It is cleared automatically after the reset operation has completed in all of the DWC_gmac clock domains. Before reprogramming any register of the DWC_gmac, you should read a zero (0) value in this bit . Note: * The Software reset function is driven only by this bit. Bit 0 of Register 64 (Channel 1 Bus Mode Register) or Register 128 (Channel 2 Bus Mode Register) has no impact on the Software reset function. * The reset operation is completed only when all resets in all active clock domains are de-asserted. Therefore, it is essential that all the PHY inputs clocks (applicable for the selected PHY interface) are present for the software reset completion. **Value** **Description** 0x0 CLEARRST 0x1 RESET	RW	0x1

dmagrp_transmit_poll_demand

Register 1 (Transmit Poll Demand Register)

The Transmit Poll Demand register enables the Tx DMA to check whether or not the DMA owns the current descriptor. The Transmit Poll Demand command is given to wake up the Tx DMA if it is in the Suspend mode. The Tx DMA can go into the Suspend mode because of an Underflow error in a transmitted frame or the unavailability of descriptors owned by it. You can give this command anytime and the Tx DMA resets this command when it again starts fetching the current descriptor from host memory. When this register is read, it always returns zero.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF801004

Module Instance	Base Address	Register Address
i_emac_emac1	0xFF802000	0xFF803004
i_emac_emac2	0xFF804000	0xFF805004

Offset: 0x1004

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
tpd RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
tpd RW 0x0															

dmagrp_transmit_poll_demand Fields

Bit	Name	Description	Access	Reset
31:0	tpd	Transmit Poll Demand When these bits are written with any value, the DMA reads the current descriptor pointed to by Register 18 (Current Host Transmit Descriptor Register). If that descriptor is not available (owned by the Host), the transmission returns to the Suspend state and the Bit 2 (TU) of Register 5 (Status Register) is asserted. If the descriptor is available, the transmission resumes.	RW	0x0

dmagrp_receive_poll_demand

Register 2 (Receive Poll Demand Register)

The Receive Poll Demand register enables the receive DMA to check for new descriptors. This command is used to wake up the Rx DMA from the SUSPEND state. The RxDMA can go into the SUSPEND state only because of the unavailability of descriptors it owns. When this register is read, it always returns zero.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF801008
i_emac_emac1	0xFF802000	0xFF803008
i_emac_emac2	0xFF804000	0xFF805008

Offset: 0x1008

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
rpd RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rpd RW 0x0															

dmagrp_receive_poll_demand Fields

Bit	Name	Description	Access	Reset
31:0	rpd	Receive Poll Demand When these bits are written with any value, the DMA reads the current descriptor pointed to by Register 19 (Current Host Receive Descriptor Register). If that descriptor is not available (owned by the Host), the reception returns to the Suspended state and the Bit 7 (RU) of Register 5 (Status Register) is not asserted. If the descriptor is available, the Rx DMA returns to the active state.	RW	0x0

dmagrp_receive_descriptor_list_address

Register 3 (Receive Descriptor List Address Register)

The Receive Descriptor List Address register points to the start of the Receive Descriptor List. The descriptor lists reside in the host's physical memory space and must be Word, Dword, or Lword-aligned (for 32-bit, 64-bit, or 128-bit data bus). The DMA internally converts it to bus width aligned address by making the corresponding LS bits low. Writing to this register is permitted only when

reception is stopped. When stopped, this register must be written to before the receive Start command is given.

You can write to this register only when Rx DMA has stopped, that is, Bit 1 (SR) is set to zero in Register 6 (Operation Mode Register). When stopped, this register can be written with a new descriptor list address. When you set the SR bit to 1, the DMA takes the newly programmed descriptor base address.

If this register is not changed when the SR bit is set to 0, then the DMA takes the descriptor address where it was stopped earlier.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF80100C
i_emac_emac1	0xFF802000	0xFF80300C
i_emac_emac2	0xFF804000	0xFF80500C

Offset: 0x100C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
rdesla_32bit RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rdesla_32bit RW 0x0														Reserved	

dmagrp_receive_descriptor_list_address Fields

Bit	Name	Description	Access	Reset
31:2	rdesla_32bit	This field contains the base address of the first descriptor in the Receive Descriptor list. The LSB bits (1:0) are ignored (32-bit wide bus) and internally taken as all-zero by the DMA. Therefore, these LSB bits are read-only (RO).	RW	0x0

dmagrp_transmit_descriptor_list_address

Register 4 (Transmit Descriptor List Address Register)

The Transmit Descriptor List Address register points to the start of the Transmit Descriptor List. The descriptor lists reside in the host's physical memory space and must be Word, Dword, or Lword-aligned (for 32-bit, 64-bit, or 128-bit data bus). The DMA internally converts it to bus width aligned address by making the corresponding LSB to low.

You can write to this register only when the Tx DMA has stopped, that is, Bit 13 (ST) is set to zero in Register 6 (Operation Mode Register). When stopped, this register can be written with a new descriptor list address. When you set the ST bit to 1, the DMA takes the newly programmed descriptor base address.

If this register is not changed when the ST bit is set to 0, then the DMA takes the descriptor address where it was stopped earlier.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF801010
i_emac_emac1	0xFF802000	0xFF803010
i_emac_emac2	0xFF804000	0xFF805010

Offset: 0x1010

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
tdesla_32bit RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
tdesla_32bit RW 0x0														Reserved	

dmagrp_transmit_descriptor_list_address Fields

Bit	Name	Description	Access	Reset
31:2	tdesla_32bit	This field contains the base address of the first descriptor in the Transmit Descriptor list. The LSB bits (1:0) are ignored (32-bit wide bus) and are internally taken as all-zero by the DMA. Therefore, these LSB bits are read-only (RO).	RW	0x0

dmagrp_status

Register 5 (Status Register)

The Status register contains all status bits that the DMA reports to the host. The Software driver reads this register during an interrupt service routine or polling. Most of the fields in this register cause the host to be interrupted. The bits of this register are not cleared when read. Writing 1'b1 to (unreserved) Bits[16:0] of this register clears these bits and writing 1'b0 has no effect. Each field (Bits[16:0]) can be masked by masking the appropriate bit in Register 7 (Interrupt Enable Register).

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF801014
i_emac_emac1	0xFF802000	0xFF803014
i_emac_emac2	0xFF804000	0xFF805014

Offset: 0x1014

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
reserved_31 RO 0x0	glpii RO 0x0	tti RO 0x0	gpi RO 0x0	gmi RO 0x0	gli RO 0x0	eb RO 0x0			ts RO 0x0			rs RO 0x0			nis RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ais RW 0x0	eri RW 0x0	fbi RW 0x0	reserved_12_11 RO 0x0		eti RW 0x0	rwt RW 0x0	rps RW 0x0	ru RW 0x0	ri RW 0x0	unf RW 0x0	ovf RW 0x0	tjt RW 0x0	tu RW 0x0	tps RW 0x0	ti RW 0x0

dmagrp_status Fields

Bit	Name	Description	Access	Reset
31	reserved_31	Reserved	RO	0x0
30	glpii	GMAC LPI Interrupt (for Channel 0) This bit indicates an interrupt event in the LPI logic of the DWC_gmac. To reset this bit to 1'b0, the software must read the corresponding registers in the DWC_gmac to get the exact cause of the interrupt and clear its source. Note: GLPII status is given only in Channel 0 DMA register and is applicable only when the Energy Efficient Ethernet feature is enabled. Otherwise, this bit is reserved.When this bit is high, the interrupt signal from the MAC (sbd_intr_o) is high. **Value** **Description** 0x0 NOINTERRUP 0x1 INTERRUP	RO	0x0

Bit	Name	Description	Access	Reset
29	tti	Timestamp Trigger Interrupt This bit indicates an interrupt event in the Timestamp Generator block of DWC_gmac. The software must read the corresponding registers in the DWC_gmac to get the exact cause of interrupt and clear its source to reset this bit to 1'b0. When this bit is high, the interrupt signal from the DWC_gmac subsystem (sbd_intr_o) is high. This bit is applicable only when the IEEE 1588 Timestamp feature is enabled. Otherwise, this bit is reserved. **Value** **Description** 0x0 NOINTERRUP 0x1 INTERRUP	RO	0x0
28	gpi	GMAC PMT Interrupt This bit indicates an interrupt event in the PMT module of the DWC_gmac. The software must read the PMT Control and Status Register in the MAC to get the exact cause of interrupt and clear its source to reset this bit to 1'b0. The interrupt signal from the DWC_gmac subsystem (sbd_intr_o) is high when this bit is high. This bit is applicable only when the Power Management feature is enabled. Otherwise, this bit is reserved. Note: The GPI and pmt_intr_o interrupts are generated in different clock domains.	RO	0x0

Bit	Name	Description	Access	Reset
27	gmi	GMAC MMC Interrupt This bit reflects an interrupt event in the MMC module of the DWC_gmac. The software must read the corresponding registers in the DWC_gmac to get the exact cause of interrupt and clear the source of interrupt to make this bit as 1'b0. The interrupt signal from the DWC_gmac subsystem (sbd_intr_o) is high when this bit is high. This bit is applicable only when the MAC Management Counters (MMC) are enabled. Otherwise, this bit is reserved. **Value** **Description** 0x0 NOINTERRUP 0x1 INTERRUP	RO	0x0

Bit	Name	Description	Access	Reset
26	gli	GMAC Line interface Interrupt When set, this bit reflects any of the following interrupt events in the DWC_gmac interfaces (if present and enabled in your configuration): * PCS (TBI, RTBI, or SGMII): Link change or auto-negotiation complete event * SMII or RGMII: Link change event * General Purpose Input Status (GPIS): Any LL or LH event on the gpi_i input ports To identify the exact cause of the interrupt, the software must first read Bit 11 and Bits[2:0] of Register 14 (Interrupt Status Register) and then to clear the source of interrupt (which also clears the GLI interrupt), read any of the following corresponding registers: * PCS (TBI, RTBI, or SGMII): Register 49 (AN Status Register) * SMII or RGMII: Register 54 (SGMII/RGMII/SMII Status Register) * General Purpose Input (GPI): Register 56 (General Purpose IO Register) The interrupt signal from the DWC_gmac subsystem (sbd_intr_o) is high when this bit is high. **Value** **Description** 0x0 NOINTERRUP 0x1 INTERRUP	RO	0x0

Bit	Name	Description	Access	Reset
25:23	eb	Error Bits This field indicates the type of error that caused a Bus Error, for example, error response on the AHB or AXI interface. This field is valid only when Bit 13 (FBI) is set. This field does not generate an interrupt. * 0 0 0: Error during Rx DMA Write Data Transfer * 0 1 1: Error during Tx DMA Read Data Transfer * 1 0 0: Error during Rx DMA Descriptor Write Access * 1 0 1: Error during Tx DMA Descriptor Write Access * 1 1 0: Error during Rx DMA Descriptor Read Access * 1 1 1: Error during Tx DMA Descriptor Read Access Note: 001 and 010 are reserved.	RO	0x0

Bit	Name	Description	Access	Reset
22:20	ts	Transmit Process State This field indicates the Transmit DMA FSM state. This field does not generate an interrupt. * 3'b000: Stopped; Reset or Stop Transmit Command issued * 3'b001: Running; Fetching Transmit Transfer Descriptor * 3'b010: Running; Waiting for status * 3'b011: Running; Reading Data from host memory buffer and queuing it to transmit buffer (Tx FIFO) * 3'b100: TIME_STAMP write state * 3'b101: Reserved for future use * 3'b110: Suspended; Transmit Descriptor Unavailable or Transmit Buffer Underflow * 3'b111: Running; Closing Transmit Descriptor **Value** **Description** 0x0 STOPPED 0x1 RUNFETCH 0x2 RUNWAIT 0x3 RUNREAD 0x4 TIMESTMP 0x5 RESERVE 0x6 SUSPTX 0x7 RUNCLOSE	RO	0x0

Bit	Name	Description	Access	Reset
19:17	rs	Received Process State This field indicates the Receive DMA FSM state. This field does not generate an interrupt. * 3'b000: Stopped: Reset or Stop Receive Command issued * 3'b001: Running: Fetching Receive Transfer Descriptor * 3'b010: Reserved for future use * 3'b011: Running: Waiting for receive packet * 3'b100: Suspended: Receive Descriptor Unavailable * 3'b101: Running: Closing Receive Descriptor * 3'b110: TIME_STAMP write state * 3'b111: Running: Transferring the receive packet data from receive buffer to host memory **Value** **Description** 0x0 STOPPED 0x1 RUNFETCH 0x2 RESERVE 0x3 RUNWAIT 0x4 SUSPEND 0x5 RUNCLOSE 0x6 TIMESTMP 0x7 RUNTRANS	RO	0x0

Bit	Name	Description	Access	Reset
16	nis	Normal Interrupt Summary Normal Interrupt Summary bit value is the logical OR of the following when the corresponding interrupt bits are enabled in Register 7 (Interrupt Enable Register): * Register 5[0]: Transmit Interrupt * Register 5[2]: Transmit Buffer Unavailable * Register 5[6]: Receive Interrupt * Register 5[14]: Early Receive Interrupt Only unmasked bits (interrupts for which interrupt enable is set in Register 7) affect the Normal Interrupt Summary bit. This is a sticky bit and must be cleared (by writing 1 to this bit) each time a corresponding bit, which causes NIS to be set, is cleared.	RW	0x0

Bit	Name	Description	Access	Reset
15	ais	Abnormal Interrupt Summary Abnormal Interrupt Summary bit value is the logical OR of the following when the corresponding interrupt bits are enabled in Register 7 (Interrupt Enable Register): * Register 5[1]: Transmit Process Stopped * Register 5[3]: Transmit Jabber Timeout * Register 5[4]: Receive FIFO Overflow * Register 5[5]: Transmit Underflow * Register 5[7]: Receive Buffer Unavailable * Register 5[8]: Receive Process Stopped * Register 5[9]: Receive Watchdog Timeout * Register 5[10]: Early Transmit Interrupt * Register 5[13]: Fatal Bus Error Only unmasked bits affect the Abnormal Interrupt Summary bit. This is a sticky bit and must be cleared each time a corresponding bit, which causes AIS to be set, is cleared.	RW	0x0
14	eri	Early Receive Interrupt This bit indicates that the DMA filled the first data buffer of the packet. This bit is cleared when the software writes 1 to this bit or Bit 6 (RI) of this register is set (whichever occurs earlier).	RW	0x0
13	fbi	Fatal Bus Error Interrupt This bit indicates that a bus error occurred, as described in Bits[25:23]. When this bit is set, the corresponding DMA engine disables all of its bus accesses.	RW	0x0
12:11	reserved_12_11	Reserved	RO	0x0

Bit	Name	Description	Access	Reset
10	eti	Early Transmit Interrupt This bit indicates that the frame to be transmitted is fully transferred to the MTL Transmit FIFO.	RW	0x0
9	rwt	Receive Watchdog Timeout When set, this bit indicates that the Receive Watchdog Timer expired while receiving the current frame and the current frame is truncated after the watchdog timeout.	RW	0x0
8	rps	Receive Process Stopped This bit is asserted when the Receive Process enters the Stopped state.	RW	0x0
7	ru	Receive Buffer Unavailable This bit indicates that the host owns the Next Descriptor in the Receive List and the DMA cannot acquire it. The Receive Process is suspended. To resume processing Receive descriptors, the host should change the ownership of the descriptor and issue a Receive Poll Demand command. If no Receive Poll Demand is issued, the Receive Process resumes when the next recognized incoming frame is received. This bit is set only when the previous Receive Descriptor is owned by the DMA.	RW	0x0
6	ri	Receive Interrupt This bit indicates that the frame reception is complete. When reception is complete, the Bit 31 of RDES1 (Disable Interrupt on Completion) is reset in the last Descriptor, and the specific frame status information is updated in the descriptor. The reception remains in the Running state.	RW	0x0

Bit	Name	Description	Access	Reset
5	unf	Transmit Underflow This bit indicates that the Transmit Buffer had an Underflow during frame transmission. Transmission is suspended and an Underflow Error TDES0[1] is set.	RW	0x0
4	ovf	Receive Overflow This bit indicates that the Receive Buffer had an Overflow during frame reception. If the partial frame is transferred to the application, the overflow status is set in RDES0[11].	RW	0x0
3	tjt	Transmit Jabber Timeout This bit indicates that the Transmit Jabber Timer expired, which happens when the frame size exceeds 2,048 (10,240 bytes when the Jumbo frame is enabled). When the Jabber Timeout occurs, the transmission process is aborted and placed in the Stopped state. This causes the Transmit Jabber Timeout TDES0[14] flag to assert.	RW	0x0
2	tu	Transmit Buffer Unavailable This bit indicates that the host owns the Next Descriptor in the Transmit List and the DMA cannot acquire it. Transmission is suspended. Bits[22:20] explain the Transmit Process state transitions. To resume processing Transmit descriptors, the host should change the ownership of the descriptor by setting TDES0[31] and then issue a Transmit Poll Demand command.	RW	0x0
1	tps	Transmit Process Stopped This bit is set when the transmission is stopped.	RW	0x0

Bit	Name	Description	Access	Reset
0	ti	Transmit Interrupt This bit indicates that the frame transmission is complete. When transmission is complete, Bit 31 (OWN) of TDES0 is reset, and the specific frame status information is updated in the descriptor.	RW	0x0

dmagrp_operation_mode

Register 6 (Operation Mode Register)

The Operation Mode register establishes the Transmit and Receive operating modes and commands. This register should be the last CSR to be written as part of the DMA initialization. This register is also present in the GMAC-MTL configuration with unused and reserved bits 24, 13, 2, and 1.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF801018
i_emac_emac1	0xFF802000	0xFF803018
i_emac_emac2	0xFF804000	0xFF805018

Offset: 0x1018

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
reserved_31_27 RO 0x0					dt RW 0x0	rsf RW 0x0	dff RW 0x0	rfa_2 RO 0x0	rfd_2 RO 0x0	tsf RW 0x0	ftf RW 0x0	reserved_19_17 RO 0x0			ttc RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ttc RW 0x0		st RW 0x0	rfd RO 0x0		rfa RO 0x0		efc RO 0x0	fef RW 0x0	fuf RW 0x0	dgf RW 0x0	rtc RW 0x0		osf RW 0x0	sr RW 0x0	reserved_0 RO 0x0

dmagrp_operation_mode Fields

Bit	Name	Description	Access	Reset
31:27	reserved_31_27	Reserved	RO	0x0
26	dt	Disable Dropping of TCP/IP Checksum Error Frames When this bit is set, the MAC does not drop the frames which only have errors detected by the Receive Checksum Offload engine. Such frames do not have any errors (including FCS error) in the Ethernet frame received by the MAC but have errors only in the encapsulated payload. When this bit is reset, all error frames are dropped if the FEF bit is reset. If the IPC Full Checksum Offload Engine (Type 2) is disabled, this bit is reserved (RO with value 1'b0). **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
25	rsf	Receive Store and Forward When this bit is set, the MTL reads a frame from the Rx FIFO only after the complete frame has been written to it, ignoring the RTC bits. When this bit is reset, the Rx FIFO operates in the cut-through mode, subject to the threshold specified by the RTC bits. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
24	dff	Disable Flushing of Received Frames When this bit is set, the Rx DMA does not flush any frames because of the unavailability of receive descriptors or buffers as it does normally when this bit is reset. This bit is reserved (and RO) in the GMAC-MTL configuration. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
23	rfa_2	MSB of Threshold for Activating Flow Control If the DWC_gmac is configured for an Rx FIFO depth of 8 KB or more, this bit (when set) provides additional threshold levels for activating the flow control in both half-duplex and full-duplex modes. This bit (as Most Significant Bit) along with the RFA (Bits[10:9]) gives the following thresholds for activating flow control: * 100: Full minus 5 KB, that is, FULL - 5KB * 101: Full minus 6 KB, that is, FULL - 6KB * 110: Full minus 7 KB, that is, FULL - 7KB * 111: Reserved This bit is reserved (and RO) if the Rx FIFO is 4 KB or less deep.	RO	0x0

Bit	Name	Description	Access	Reset
22	rfd_2	MSB of Threshold for Deactivating Flow Control If the DWC_gmac is configured for Rx FIFO size of 8 KB or more, this bit (when set) provides additional threshold levels for deactivating the flow control in both half-duplex and full-duplex modes. This bit (as Most Significant Bit) along with the RFD (Bits[12:11]) gives the following thresholds for deactivating flow control: * 100: Full minus 5 KB, that is, FULL - 5KB * 101: Full minus 6 KB, that is, FULL - 6KB * 110: Full minus 7 KB, that is, FULL - 7KB * 111: Reserved This bit is reserved (and RO) if the Rx FIFO is 4 KB or less deep.	RO	0x0
21	tsf	Transmit Store and Forward When this bit is set, transmission starts when a full frame resides in the MTL Transmit FIFO. When this bit is set, the TTC values specified in Bits[16:14] are ignored. This bit should be changed only when the transmission is stopped. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
20	ftf	Flush Transmit FIFO When this bit is set, the transmit FIFO controller logic is reset to its default values and thus all data in the Tx FIFO is lost or flushed. This bit is cleared internally when the flushing operation is completed. The Operation Mode register should not be written to until this bit is cleared. The data which is already accepted by the MAC transmitter is not flushed. It is scheduled for transmission and results in underflow and runt frame transmission. Note: The flush operation is complete only when the Tx FIFO is emptied of its contents and all the pending Transmit Status of the transmitted frames are accepted by the host. To complete this flush operation, the PHY transmit clock (clk_tx_i) is required to be active. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
19:17	reserved_19_17	Reserved	RO	0x0

Bit	Name	Description	Access	Reset
16:14	ttc	Transmit Threshold Control These bits control the threshold level of the MTL Transmit FIFO. Transmission starts when the frame size within the MTL Transmit FIFO is larger than the threshold. In addition, full frames with a length less than the threshold are also transmitted. These bits are used only when Bit 21 (TSF) is reset. * 000: 64 * 001: 128 * 010: 192 * 011: 256 * 100: 40 * 101: 32 * 110: 24 * 111: 16 **Value** **Description** 0x0 TTCTHESH64 0x1 TTCTHRES128 0x2 TTCTHRES192 0x3 TTCTHRES256 0x4 TTCTHRES40 0x5 TTCTHRES32 0x6 TTCTHRES24 0x7 TTCTHRES16	RW	0x0

Bit	Name	Description	Access	Reset
13	st	Start or Stop Transmission Command When this bit is set, transmission is placed in the Running state, and the DMA checks the Transmit List at the current position for a frame to be transmitted. Descriptor acquisition is attempted either from the current position in the list, which is the Transmit List Base Address set by Register 4 (Transmit Descriptor List Address Register), or from the position retained when transmission was stopped previously. If the DMA does not own the current descriptor, transmission enters the Suspended state and Bit 2 (Transmit Buffer Unavailable) of Register 5 (Status Register) is set. The Start Transmission command is effective only when transmission is stopped. If the command is issued before setting Register 4 (Transmit Descriptor List Address Register), then the DMA behavior is unpredictable. When this bit is reset, the transmission process is placed in the Stopped state after completing the transmission of the current frame. The Next Descriptor position in the Transmit List is saved, and it becomes the current position when transmission is restarted. To change the list address, you need to program Register 4 (Transmit Descriptor List Address Register) with a new value when this bit is reset. The new value is considered when this bit is set again. The stop transmission command is effective only when the transmission of the current frame is complete or the transmission is in the Suspended state. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
12:11	rfd	Threshold for Deactivating Flow Control (in half-duplex and full-duplex) These bits control the threshold (Fill-level of Rx FIFO) at which the flow control is de-asserted after activation. - 00: Full minus 1 KB, that is, FULL - 1KB - 01: Full minus 2 KB, that is, FULL - 2KB - 10: Full minus 3 KB, that is, FULL - 3KB - 11: Full minus 4 KB, that is, FULL - 4KB The de-assertion is effective only after flow control is asserted. If the Rx FIFO is 8 KB or more, an additional bit (RFD_2) is used for more threshold levels as described in Bit 22. These bits are reserved and read-only when the Rx FIFO depth is less than 4 KB. Note: For proper flow control, the value programmed in the "RFD_2, RFD" fields should be equal to or more than the value programmed in the "RFA_2, RFA" fields. **Value** — **Description** 0x0 — FIFOFULL_1K 0x1 — FIFOFULL_2K 0x2 — FIFOFULL_3K 0x3 — FIFOFULL_4K	RO	0x0

Bit	Name	Description	Access	Reset
10:9	rfa	Threshold for Activating Flow Control (in half-duplex and full-duplex) These bits control the threshold (Fill level of Rx FIFO) at which the flow control is activated. - 00: Full minus 1 KB, that is, FULL - 1KB - 01: Full minus 2 KB, that is, FULL - 2KB - 10: Full minus 3 KB, that is, FULL - 3KB - 11: Full minus 4 KB, that is, FULL - 4KB These values are applicable only to Rx FIFOs of 4 KB or more and when Bit 8 (EFC) is set high. If the Rx FIFO is 8 KB or more, an additional Bit (RFA_2) is used for more threshold levels as described in Bit 23. These bits are reserved and read-only when the depth of Rx FIFO is less than 4 KB. Note: When FIFO size is exactly 4 KB, although the DWC_gmac allows you to program the value of these bits to 11, the software should not program these bits to 2'b11. The value 2'b11 means flow control on FIFO empty condition. **Value** **Description** 0x0 FIFOFULL_1K 0x1 FIFOFULL_2K 0x2 FIFOFULL_3K 0x3 FIFOFULL_4K	RO	0x0
8	efc	Reserved **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RO	0x0

Bit	Name	Description	Access	Reset
7	fef	Forward Error Frames When this bit is reset, the Rx FIFO drops frames with error status (CRC error, collision error, GMII_ER, giant frame, watchdog timeout, or overflow). However, if the start byte (write) pointer of a frame is already transferred to the read controller side (in Threshold mode) , then the frame is not dropped. In the GMAC-MTL configuration in which the Frame Length FIFO is also enabled during core configuration, the Rx FIFO drops the error frames if that frame's start byte is not transferred (output) on the ARI bus. When the FEF bit is set, all frames except runt error frames are forwarded to the DMA. If the Bit 25 (RSF) is set and the Rx FIFO overflows when a partial frame is written, then the frame is dropped irrespective of the FEF bit setting. However, if the Bit 25 (RSF) is reset and the Rx FIFO overflows when a partial frame is written, then a partial frame may be forwarded to the DMA. Note: When FEF bit is reset, the giant frames are dropped if the giant frame status is given in Rx Status in the following configura-tions: * The IP checksum engine (Type 1) and full checksum offload engine (Type 2) are not selected. * The advanced timestamp feature is not selected but the extended status is selected. The extended status is available with the following features: - L3-L4 filter in GMAC-CORE or GMAC-MTL configurations - Full checksum offload engine (Type 2) with enhanced descriptor format in the GMAC-DMA, GMAC-AHB, or GMAC-AXI configurations. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
6	fuf	Forward Undersized Good Frames When set, the Rx FIFO forwards Undersized frames (frames with no Error and length less than 64 bytes) including pad-bytes and CRC. When reset, the Rx FIFO drops all frames of less than 64 bytes, unless a frame is already transferred because of the lower value of Receive Threshold, for example, RTC = 01. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
5	dgf	Drop Giant Frames When set, the MAC drops the received giant frames in the Rx FIFO, that is, frames that are larger than the computed giant frame limit. When reset, the MAC does not drop the giant frames in the Rx FIFO. Note: This bit is available in the following configurations in which the giant frame status is not provided in Rx status and giant frames are not dropped by default: * Configurations in which IP Checksum Offload (Type 1) is selected in Rx * Configurations in which the IPC Full Checksum Offload Engine (Type 2) is selected in Rx with normal descriptor format * Configurations in which the Advanced Timestamp feature is selected In all other configurations, this bit is not used (reserved and always reset).	RW	0x0

Bit	Name	Description	Access	Reset
4:3	rtc	Receive Threshold Control These two bits control the threshold level of the MTL Receive FIFO. Transfer (request) to DMA starts when the frame size within the MTL Receive FIFO is larger than the threshold. In addition, full frames with length less than the threshold are transferred automatically. The value of 11 is not applicable if the configured Receive FIFO size is 128 bytes. These bits are valid only when the RSF bit is zero, and are ignored when the RSF bit is set to 1. * 00: 64 * 01: 32 * 10: 96 * 11: 128 **Value** **Description** 0x0 THRFIFO64 0x1 THRFIFO32 0x2 THRFIFO96 0x3 THRFIFO128	RW	0x0
2	osf	Operate on Second Frame When this bit is set, it instructs the DMA to process the second frame of the Transmit data even before the status for the first frame is obtained. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
1	sr	Start or Stop Receive When this bit is set, the Receive process is placed in the Running state. The DMA attempts to acquire the descriptor from the Receive list and processes the incoming frames. The descriptor acquisition is attempted from the current position in the list, which is the address set by Register 3 (Receive Descriptor List Address Register) or the position retained when the Receive process was previously stopped. If the DMA does not own the descriptor, reception is suspended and Bit 7 (Receive Buffer Unavailable) of Register 5 (Status Register) is set. The Start Receive command is effective only when the reception has stopped. If the command is issued before setting Register 3 (Receive Descriptor List Address Register), the DMA behavior is unpredictable. When this bit is cleared, the Rx DMA operation is stopped after the transfer of the current frame. The next descriptor position in the Receive list is saved and becomes the current position after the Receive process is restarted. The Stop Receive command is effective only when the Receive process is in either the Running (waiting for receive packet) or in the Suspended state. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
0	reserved_0	Reserved	RO	0x0

dmagrp_interrupt_enable

Register 7 (Interrupt Enable Register)

The Interrupt Enable register enables the interrupts reported by Register 5 (Status Register). Setting a bit to 1'b1 enables a corresponding interrupt. After a hardware or software reset, all interrupts are disabled.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF80101C
i_emac_emac1	0xFF802000	0xFF80301C
i_emac_emac2	0xFF804000	0xFF80501C

Offset: 0x101C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
reserved_31_17 RO 0x0															nie RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
aie RW 0x0	ere RW 0x0	fbe RW 0x0	reserved_12_11 RO 0x0		ete RW 0x0	rwe RW 0x0	rse RW 0x0	rue RW 0x0	rie RW 0x0	une RW 0x0	ove RW 0x0	tje RW 0x0	tue RW 0x0	tse RW 0x0	tie RW 0x0

dmagrp_interrupt_enable Fields

Bit	Name	Description	Access	Reset
31:17	reserved_31_17	Reserved	RO	0x0

Bit	Name	Description	Access	Reset
16	nie	Normal Interrupt Summary Enable When this bit is set, normal interrupt summary is enabled. When this bit is reset, normal interrupt summary is disabled. This bit enables the following interrupts in Register 5 (Status Register): * Register 5[0]: Transmit Interrupt * Register 5[2]: Transmit Buffer Unavailable * Register 5[6]: Receive Interrupt * Register 5[14]: Early Receive Interrupt **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
15	aie	Abnormal Interrupt Summary Enable When this bit is set, abnormal interrupt summary is enabled. When this bit is reset, the abnormal interrupt summary is disabled. This bit enables the following interrupts in Register 5 (Status Register): * Register 5[1]: Transmit Process Stopped * Register 5[3]: Transmit Jabber Timeout * Register 5[4]: Receive Overflow * Register 5[5]: Transmit Underflow * Register 5[7]: Receive Buffer Unavailable * Register 5[8]: Receive Process Stopped * Register 5[9]: Receive Watchdog Timeout * Register 5[10]: Early Transmit Interrupt * Register 5[13]: Fatal Bus Error **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
14	ere	Early Receive Interrupt Enable When this bit is set with Normal Interrupt Summary Enable (Bit 16), the Early Receive Interrupt is enabled. When this bit is reset, the Early Receive Interrupt is disabled. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
13	fbe	Fatal Bus Error Enable When this bit is set with Abnormal Interrupt Summary Enable (Bit 15), the Fatal Bus Error Interrupt is enabled. When this bit is reset, the Fatal Bus Error Enable Interrupt is disabled. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
12:11	reserved_12_11	Reserved	RO	0x0
10	ete	Early Transmit Interrupt Enable When this bit is set with an Abnormal Interrupt Summary Enable (Bit 15), the Early Transmit Interrupt is enabled. When this bit is reset, the Early Transmit Interrupt is disabled. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
9	rwe	Receive Watchdog Timeout Enable When this bit is set with Abnormal Interrupt Summary Enable (Bit 15), the Receive Watchdog Timeout Interrupt is enabled. When this bit is reset, the Receive Watchdog Timeout Interrupt is disabled. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
8	rse	Receive Stopped Enable When this bit is set with Abnormal Interrupt Summary Enable (Bit 15), the Receive Stopped Interrupt is enabled. When this bit is reset, the Receive Stopped Interrupt is disabled. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
7	rue	Receive Buffer Unavailable Enable When this bit is set with Abnormal Interrupt Summary Enable (Bit 15), the Receive Buffer Unavailable Interrupt is enabled. When this bit is reset, the Receive Buffer Unavailable Interrupt is disabled.	RW	0x0
6	rie	Receive Interrupt Enable When this bit is set with Normal Interrupt Summary Enable (Bit 16), the Receive Interrupt is enabled. When this bit is reset, the Receive Interrupt is disabled. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
5	une	Underflow Interrupt Enable When this bit is set with Abnormal Interrupt Summary Enable (Bit 15), the Transmit Underflow Interrupt is enabled. When this bit is reset, the Underflow Interrupt is disabled. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
4	ove	Overflow Interrupt Enable When this bit is set with Abnormal Interrupt Summary Enable (Bit 15), the Receive Overflow Interrupt is enabled. When this bit is reset, the Overflow Interrupt is disabled. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
3	tje	Transmit Jabber Timeout Enable When this bit is set with Abnormal Interrupt Summary Enable (Bit 15), the Transmit Jabber Timeout Interrupt is enabled. When this bit is reset, the Transmit Jabber Timeout Interrupt is disabled. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
2	tue	Transmit Buffer Unavailable Enable When this bit is set with Normal Interrupt Summary Enable (Bit 16), the Transmit Buffer Unavailable Interrupt is enabled. When this bit is reset, the Transmit Buffer Unavailable Interrupt is disabled. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
1	tse	Transmit Stopped Enable When this bit is set with Abnormal Interrupt Summary Enable (Bit 15), the Transmission Stopped Interrupt is enabled. When this bit is reset, the Transmission Stopped Interrupt is disabled. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
0	tie	Transmit Interrupt Enable When this bit is set with Normal Interrupt Summary Enable (Bit 16), the Transmit Interrupt is enabled. When this bit is reset, the Transmit Interrupt is disabled. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0

dmagrp_missed_frame_and_buffer_overflow_counter

Register 8 (Missed Frame and Buffer Overflow Counter Register)

The DMA maintains two counters to track the number of frames missed during reception. This register reports the current value of the counter. The counter is used for diagnostic purposes. Bits[15:0]

indicate missed frames because of the host buffer being unavailable. Bits[27:17] indicate missed frames because of buffer overflow conditions (MTL and MAC) and runt frames (good frames of less than 64 bytes) dropped by the MTL.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF801020
i_emac_emac1	0xFF802000	0xFF803020
i_emac_emac2	0xFF804000	0xFF805020

Offset: 0x1020

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
reserved_31_29 RO 0x0			ovfcntovf RO 0x0	ovffrmcnt RO 0x0											miscntovf RO 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
misfrmcnt RO 0x0															

dmagrp_missed_frame_and_buffer_overflow_counter Fields

Bit	Name	Description	Access	Reset
31:29	reserved_31_29	Reserved	RO	0x0
28	ovfcntovf	Overflow Bit for FIFO Overflow Counter This bit is set every time the Overflow Frame Counter (Bits[27:17]) overflows, that is, the Rx FIFO overflows with the overflow frame counter at maximum value. In such a scenario, the overflow frame counter is reset to all-zeros and this bit indicates that the rollover happened.	RO	0x0

Bit	Name	Description	Access	Reset
27:17	ovffrmcnt	Overflow Frame Counter This field indicates the number of frames missed by the application. This counter is incremented each time the MTL FIFO overflows. The counter is cleared when this register is read with mci_be_i[2] at 1'b1.	RO	0x0
16	miscntovf	Overflow Bit for Missed Frame Counter This bit is set every time Missed Frame Counter (Bits[15:0]) overflows, that is, the DMA discards an incoming frame because of the Host Receive Buffer being unavailable with the missed frame counter at maximum value. In such a scenario, the Missed frame counter is reset to all-zeros and this bit indicates that the rollover happened.	RO	0x0
15:0	misfrmcnt	Missed Frame Counter This field indicates the number of frames missed by the controller because of the Host Receive Buffer being unavailable. This counter is incremented each time the DMA discards an incoming frame. The counter is cleared when this register is read with mci_be_i[0] at 1'b1.	RO	0x0

dmagrp_receive_interrupt_watchdog_timer

Register 9 (Receive Interrupt Watchdog Timer Register)

This register, when written with non-zero value, enables the watchdog timer for the Receive Interrupt (Bit 6) of Register 5 (Status Register)

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF801024

Module Instance	Base Address	Register Address
i_emac_emac1	0xFF802000	0xFF803024
i_emac_emac2	0xFF804000	0xFF805024

Offset: 0x1024

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
reserved_31_8 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved_31_8 RO 0x0								riwt RW 0x0							

dmagrp_receive_interrupt_watchdog_timer Fields

Bit	Name	Description	Access	Reset
31:8	reserved_31_8	Reserved	RO	0x0
7:0	riwt	RI Watchdog Timer Count This bit indicates the number of system clock cycles multiplied by 256 for which the watchdog timer is set. The watchdog timer gets triggered with the programmed value after the Rx DMA completes the transfer of a frame for which the RI status bit is not set because of the setting in the corresponding descriptor RDES1[31]. When the watchdog timer runs out, the RI bit is set and the timer is stopped. The watchdog timer is reset when the RI bit is set high because of automatic setting of RI as per RDES1[31] of any received frame.	RW	0x0

dmagrp_axi_bus_mode

The AXI Bus Mode Register controls the behavior of the AXI master. It is mainly used to control the burst splitting and the number of outstanding requests.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF801028
i_emac_emac1	0xFF802000	0xFF803028
i_emac_emac2	0xFF804000	0xFF805028

Offset: 0x1028

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
en_lpi RW 0x0	lpi_xit_frm RW 0x0	Reserved						wr_osr_lmt RW 0x1				rd_osr_lmt RW 0x1			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		onekbbe RW 0x0	axi_aal RO 0x0	Reserved								blen16 RW 0x0	blen8 RW 0x0	blen4 RW 0x0	undefined RO 0x1

dmagrp_axi_bus_mode Fields

Bit	Name	Description	Access	Reset
31	en_lpi	When set to 1, this bit enables the LPI mode supported by the AXI master and accepts the LPI request from the AXI System Clock controller. When set to 0, this bit disables the LPI mode and always denies the LPI request from the AXI System Clock controller. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
30	lpi_xit_frm	When set to 1, this bit enables the GMAC-AXI to come out of the LPI mode only when the Magic Packet or Remote Wake Up Packet is received. When set to 0, this bit enables the GMAC-AXI to come out of LPI mode when any frame is received. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
23:20	wr_osr_lmt	AXI Maximum Write OutStanding Request Limit	RW	0x1
19:16	rd_osr_lmt	This value limits the maximum outstanding request on the AXI read interface. Maximum outstanding requests = RD_OSR_LMT+1	RW	0x1

Bit	Name	Description	Access	Reset
13	onekbbe	1 KB Boundary Crossing Enable for the GMAC-AXI Master When set, the GMAC-AXI Master performs burst transfers that do not cross 1 KB boundary. When reset, the GMAC-AXI Master performs burst transfers that do not cross 4 KB boundary. **Value** **Description** 0x0 FOUR_K_BOUNDARY 0x1 ONE_K_BOUNDARY	RW	0x0
12	axi_aal	This bit is read-only bit and reflects the Bit 25 (AAL) of Register 0 (Bus Mode Register). When this bit is set to 1, the GMAC-AXI performs address-aligned burst transfers on both read and write channels. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RO	0x0
3	blen16	When this bit is set to 1 or UNDEFINED is set to 1, the GMAC-AXI is allowed to select a burst length of 16 on the AXI Master interface. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
2	blen8	When this bit is set to 1, the GMAC-AXI is allowed to select a burst length of 8 on the AXI Master interface. Setting this bit has no effect when UNDEFINED is set to 1. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
1	blen4	When this bit is set to 1, the GMAC-AXI is allowed to select a burst length of 4 on the AXI Master interface. Setting this bit has no effect when UNDEFINED is set to 1. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
0	undefined	This bit is read-only bit and indicates the complement (invert) value of Bit 16 (FB) in Register 0 (Bus Mode Register[16]). * When this bit is set to 1, the GMAC-AXI is allowed to perform any burst length equal to or below the maximum allowed burst length programmed in Bits[7:1]. * When this bit is set to 0, the GMAC-AXI is allowed to perform only fixed burst lengths as indicated by BLEN16, BLEN8, or BLEN4, or a burst length of 1. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RO	0x1

dmagrp_ahb_or_axi_status

Register 11 (AHB or AXI Status Register)

This register provides the active status of the AHB master interface or AXI interface's read and write channels. This register is present and valid only in the GMAC-AHB and GMAC-AXI configurations. This register is useful for debugging purposes. In addition, this register is valid only in the Channel 0 DMA when multiple channels are present in the AV mode.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF80102C
i_emac_emac1	0xFF802000	0xFF80302C
i_emac_emac2	0xFF804000	0xFF80502C

Offset: 0x102C

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
reserved_31_2 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved_31_2 RO 0x0														axirdsts RO 0x0	axwhsts RO 0x0

dmagrp_ahb_or_axi_status Fields

Bit	Name	Description	Access	Reset
31:2	reserved_31_2	Reserved	RO	0x0
1	axirdsts	AXI Master Read Channel Status When high, it indicates that AXI Master's read channel is active and transferring data.	RO	0x0

Bit	Name	Description	Access	Reset
0	axwhsts	AXI Master Write Channel or AHB Master Status When high, it indicates that AXI Master's write channel is active and transferring data in the GMAC-AXI configuration. In the GMAC-AHB configuration, it indicates that the AHB master interface FSMs are in the non-idle state.	RO	0x0

dmagrp_current_host_transmit_descriptor

Register 18 (Current Host Transmit Descriptor Register)

The Current Host Transmit Descriptor register points to the start address of the current Transmit Descriptor read by the DMA.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF801048
i_emac_emac1	0xFF802000	0xFF803048
i_emac_emac2	0xFF804000	0xFF805048

Offset: 0x1048

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
curtdesaptr RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
curtdesaptr RO 0x0															

dmagrp_current_host_transmit_descriptor Fields

Bit	Name	Description	Access	Reset
31:0	curtdesaptr	Host Transmit Descriptor Address Pointer Cleared on Reset. Pointer updated by the DMA during operation.	RO	0x0

dmagrp_current_host_receive_descriptor

Register 19 (Current Host Receive Descriptor Register)

The Current Host Receive Descriptor register points to the start address of the current Receive Descriptor read by the DMA.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF80104C
i_emac_emac1	0xFF802000	0xFF80304C
i_emac_emac2	0xFF804000	0xFF80504C

Offset: 0x104C

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
currdesaptr RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
currdesaptr RO 0x0															

dmagrp_current_host_receive_descriptor Fields

Bit	Name	Description	Access	Reset
31:0	currdesaptr	Host Receive Descriptor Address Pointer Cleared on Reset. Pointer updated by the DMA during operation.	RO	0x0

dmagrp_current_host_transmit_buffer_address

Register 20 (Current Host Transmit Buffer Address Register)

The Current Host Transmit Buffer Address register points to the current Transmit Buffer Address being read by the DMA.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF801050
i_emac_emac1	0xFF802000	0xFF803050
i_emac_emac2	0xFF804000	0xFF805050

Offset: 0x1050

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
curtbufaptr RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
curtbufaptr RO 0x0															

dmagrp_current_host_transmit_buffer_address Fields

Bit	Name	Description	Access	Reset
31:0	curtbufaptr	Host Transmit Buffer Address Pointer Cleared on Reset. Pointer updated by the DMA during operation.	RO	0x0

dmagrp_current_host_receive_buffer_address

Register 21 (Current Host Receive Buffer Address Register)

The Current Host Receive Buffer Address register points to the current Receive Buffer address being read by the DMA.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF801054
i_emac_emac1	0xFF802000	0xFF803054
i_emac_emac2	0xFF804000	0xFF805054

Offset: 0x1054

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
currbufaptr RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
currbufaptr RO 0x0															

dmagrp_current_host_receive_buffer_address Fields

Bit	Name	Description	Access	Reset
31:0	currbufaptr	Host Receive Buffer Address Pointer Cleared on Reset. Pointer updated by the DMA during operation.	RO	0x0

dmagrp_hw_feature

Register 22 (HW Feature Register)

This register indicates the presence of the optional features or functions of the DWC_gmac. The software driver can use this register to dynamically enable or disable the programs related to the optional blocks.

Module Instance	Base Address	Register Address
i_emac_emac0	0xFF800000	0xFF801058
i_emac_emac1	0xFF802000	0xFF803058
i_emac_emac2	0xFF804000	0xFF805058

Offset: 0x1058

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
reserved_31 RO 0x0	actphyif RO 0x0			savlanins RO 0x1	flexippsen RO 0x1	inttsen RO 0x1	enhdessel RO 0x1	txchcnt RO 0x0		rxchcnt RO 0x0		rxfifosize RO 0x1	rxtyp2coe RO 0x1	rxtyp1coe RO 0x0	txoesel RO 0x1
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
avsel RO 0x1	eeesel RO 0x1	tsver2sel RO 0x1	tsver1sel RO 0x0	mmcsel RO 0x1	mgksel RO 0x0	rwksel RO 0x0	smasel RO 0x1	l3l4fltren RO 0x1	pcsselRO 0x0	addmacadrsel RO 0x1	hashsel RO 0x1	exthashen RO 0x1	hdsel RO 0x1	gmiisel RO 0x1	miisel RO 0x1

dmagrp_hw_feature Fields

Bit	Name	Description	Access	Reset
31	reserved_31	Reserved	RO	0x0

Bit	Name	Description	Access	Reset
30:28	actphyif	Active or Selected PHY interface When you have multiple PHY interfaces in your configuration, this field indicates the sampled value of phy_intf_sel_i during reset de-assertion * 0000: GMII or MII * 0001: RGMII * 0010: SGMII * 0011: TBI * 0100: RMII * 0101: RTBI * 0110: SMII * 0111: RevMII * All Others: Reserved **Value** **Description** 0x0 GMIIMII0 0x1 RGMII1 0x2 SGMII2 0x3 TBI3 0x4 RMII4 0x5 RTBI5 0x6 SMII6 0x7 REVMII7	RO	0x0
27	savlanins	Source Address or VLAN Insertion	RO	0x1
26	flexippsen	Flexible Pulse-Per-Second Output	RO	0x1
25	inttsen	Timestamping with Internal System Time	RO	0x1
24	enhdessel	Alternate (Enhanced Descriptor) **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RO	0x1

Bit	Name	Description	Access	Reset
23:22	txchcnt	Number of additional Tx channels **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RO	0x0
21:20	rxchcnt	Number of additional Rx channels **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RO	0x0
19	rxfifosize	Rx FIFO > 2,048 Bytes **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RO	0x1
18	rxtyp2coe	IP Checksum Offload (Type 2) in Rx **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RO	0x1
17	rxtyp1coe	IP Checksum Offload (Type 1) in Rx Note: If IPCHKSUM_EN = Enabled and IPC_FULL_OFFLOAD = Enabled, then RXTYP1COE = 0 and RXTYP2COE =1. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RO	0x0
16	txoesel	Checksum Offload in Tx **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RO	0x1

Bit	Name	Description	Access	Reset
15	avsel	AV Feature Value: Description 0x0: DISABLED 0x1: ENABLED	RO	0x1
14	eeesel	Energy Efficient Ethernet Value: Description 0x0: DISABLED 0x1: ENABLED	RO	0x1
13	tsver2sel	IEEE 1588-2008 Advanced Timestamp Value: Description 0x0: DISABLED 0x1: ENABLED	RO	0x1
12	tsver1sel	Only IEEE 1588-2002 Timestamp Value: Description 0x0: DISABLED 0x1: ENABLED	RO	0x0
11	mmcsel	RMON Module Value: Description 0x0: DISABLED 0x1: ENABLED	RO	0x1
10	mgksel	PMT Magic Packet Value: Description 0x0: DISABLED 0x1: ENABLED	RO	0x0

Bit	Name	Description	Access	Reset
9	rwksel	PMT Remote Wakeup Value: Description 0x0: DISABLED 0x1: ENABLED	RO	0x0
8	smasel	SMA (MDIO) Interface Value: Description 0x0: DISABLED 0x1: ENABLED	RO	0x1
7	l3l4fltren	Layer 3 and Layer 4 Filter Feature	RO	0x1
6	pcssel	PCS registers (TBI, SGMII, or RTBI PHY interface) Value: Description 0x0: DISABLED 0x1: ENABLED	RO	0x0
5	addmacadrsel	Multiple MAC Address Registers Value: Description 0x0: DISABLED 0x1: ENABLED	RO	0x1
4	hashsel	HASH Filter Value: Description 0x0: DISABLED 0x1: ENABLED	RO	0x1
3	exthashen	Expanded DA Hash Filter	RO	0x1
2	hdsel	Half-Duplex support Value: Description 0x0: DISABLED 0x1: ENABLED	RO	0x1

Bit	Name	Description	Access	Reset
1	gmiisel	1000 Mbps support Value: Description 0x0: DISABLED 0x1: ENABLED	RO	0x1
0	miisel	10 or 100 Mbps support Value: Description 0x0: DISABLED 0x1: ENABLED	RO	0x1

emac_rx_ecc Address Map

Module Instance	Base Address	End Address
ecc_emac0_rx_ecc_registerBlock	0xFF8C0800	0xFF8C0BFF
ecc_emac1_rx_ecc_registerBlock	0xFF8C1000	0xFF8C13FF
ecc_emac2_rx_ecc_registerBlock	0xFF8C1800	0xFF8C1BFF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
IP_REV_ID on page 11-38	0x0	32	RO	0x0	
CTRL on page 11-38	0x8	32	RW	0x0	ECC Control Register
INITSTAT on page 11-39	0xC	32	RW	0x0	This bit is used to set the initialize the memory and ecc to a known value
ERRINTEN on page 11-40	0x10	32	RW	0x0	Error Interrupt enable
ERRINTENS on page 11-41	0x14	32	RW	0x0	Error Interrupt set

Register	Offset	Width	Access	Reset Value	Description
ERRINTENR on page 11-42	0x18	32	RW	0x0	Error Interrupt reset.
INTMODE on page 11-43	0x1C	32	RW	0x0	Reads reflect SERRINTEN.
INTSTAT on page 11-43	0x20	32	RW	0x0	This bit is used to enable interrupt generation on SERR lookup table overflow. When all the entries in the table are valid=1 and this is bit is enabled, serr_req signal will be asserted.
INTTEST on page 11-44	0x24	32	RW	0x0	This bits is used to test interrupt from ECC RAM to GIC
MODSTAT on page 11-45	0x28	32	RW	0x0	Counter feature status flag
DERRADDRA on page 11-46	0x2C	32	RO	0x0	This register shows the address of PORTA current double-bit error. RAM size will determine the maximum number of address bits.
SERRADDRA on page 11-47	0x30	32	RO	0x0	This register shows the address of PORTA current single-bit error. RAM size will determine the maximum number of address bits.
SERRCNTREG on page 11-47	0x3C	32	RW	0x0	Maximum counter value for single-bit error interrupt
ECC_Addrbus on page 11-48	0x40	32	RW	0x0	MSB bit of address is determined by ADR.
ECC_RData0bus on page 11-49	0x44	32	RO	0x0	Data will be read to this register field.
ECC_RData1bus on page 11-50	0x48	32	RO	0x0	Data will be read to this register field.

Register	Offset	Width	Access	Reset Value	Description
ECC_RData2bus on page 11-50	0x4C	32	RO	0x0	Data will be read to this register field.
ECC_RData3bus on page 11-51	0x50	32	RO	0x0	Data will be read to this register field.
ECC_WData0bus on page 11-52	0x54	32	WO	0x0	Data from the register will be written to the RAM.
ECC_WData1bus on page 11-53	0x58	32	WO	0x0	Data from the register will be written to the RAM.
ECC_WData2bus on page 11-53	0x5C	32	WO	0x0	Data from the register will be written to the RAM.
ECC_WData3bus on page 11-54	0x60	32	WO	0x0	Data from the register will be written to the RAM.
ECC_RDataecc0bus on page 11-55	0x64	32	RO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.
ECC_RDataecc1bus on page 11-56	0x68	32	RO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.
ECC_WDataecc0bus on page 11-57	0x6C	32	WO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Register	Offset	Width	Access	Reset Value	Description
ECC_WDataecc1bus on page 11-58	0x70	32	WO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.
ECC_dbytectrl on page 11-59	0x74	32	RW	0x0	Max number of implemented byte enabled is DAT/8
ECC_accctrl on page 11-60	0x78	32	RW	0x0	These bits determine which byte of data/ecc to write to RAM.
ECC_startacc on page 11-61	0x7C	32	RW	0x0	These bits determine which byte of data/ecc to write to RAM.
ECC_wdctrl on page 11-61	0x80	32	RW	0x0	Bits to Enable/Disable Watch Dog Timer
SERRLKUPA0 on page 11-62	0x90	32	RW	0x0	Single-bit error address in LOOKUP TABLE for PORTA.

emac_rx_ecc Summary

Module Instance	Base Address
ecc_emac0_rx_ecc_registerBlock	0xFF8C0800
ecc_emac1_rx_ecc_registerBlock	0xFF8C1000
ecc_emac2_rx_ecc_registerBlock	0xFF8C1800

Register Address Offset	Bit Fields
ecc_emac0_rx_ecc_register-Block	

Register Address Offset	Bit Fields															
IP_REV_ID 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SIREV RO 0x0															
CTRL 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															INITA 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							CNT_RSTA 0x0	Reserved							ECC_EN 0x0
INITSTAT 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															INITCOM-PLETEA 0x0
ERRINTEN 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTEN 0x0
ERRINTENS 0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTS 0x0
ERRINTENR 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTR 0x0

Register Address Offset	Bit Fields															
INTMODE 0x1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															INTONCMP 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							INTONOVF 0x0	Reserved							INTMODE 0x0
INTSTAT 0x20	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							DERRPENA 0x0	Reserved							SERRPENA 0x0
INTTEST 0x24	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							TDERRA 0x0	Reserved							TSERRA 0x0
MODSTAT 0x28	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															CMPFLGA 0x0
DERRADDRA 0x2C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				Address 0x0											
SERRADDRA 0x30	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				Address 0x0											

Register Address Offset	Bit Fields															
SERRCNTREG 0x3C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	SERRCNT 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SERRCNT 0x0															
ECC_Addrbus 0x40	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				ECC_AddrBUS 0x0											
ECC_RData0bus 0x44	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_RData1bus 0x48	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													ECC_RDataBUS 0x0		
ECC_RData2bus 0x4C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_RData3bus 0x50	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															

Register Address Offset	Bit Fields															
ECC_WData0bus 0x54	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_WData1bus 0x58	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													ECC_WDataBUS 0x0		
ECC_WData2bus 0x5C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_WData3bus 0x60	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_RDataecc0bus 0x64	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_RDataecc3BUS 0x0							Reserved	ECC_RDataecc2BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_RDataecc1BUS 0x0							Reserved	ECC_RDataecc0BUS 0x0						
ECC_RDataecc1bus 0x68	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_RDataecc7BUS 0x0							Reserved	ECC_RDataecc6BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_RDataecc5BUS 0x0							Reserved	ECC_RDataecc4BUS 0x0						

Register Address Offset	Bit Fields															
ECC_WDataecc0bus 0x6C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_WDataecc3BUS 0x0							Reserved	ECC_WDataecc2BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_WDataecc1BUS 0x0							Reserved	ECC_WDataecc0BUS 0x0						
ECC_WDataecc1bus 0x70	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_WDataecc7BUS 0x0							Reserved	ECC_WDataecc6BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_WDataecc5BUS 0x0							Reserved	ECC_WDataecc4BUS 0x0						
ECC_dbytectrl 0x74	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															DBEN 0x0
ECC_accctrl 0x78	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							RDWR 0x0	Reserved						ECCOVR 0x0	DATAOVR 0x0
ECC_startacc 0x7C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															ENBUSA 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															
ECC_wdctrl 0x80	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															WDEN_RAM 0x0

Register Address Offset	Bit Fields															
SERRLKUPA0 0x90	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	VALID RW 0x0	Reserved														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				Address RO 0x0											
ecc_emac1_rx_ecc_register-Block																
IP_REV_ID 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SIREV RO 0x0															
CTRL 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															INITA 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							CNT_RSTA 0x0	Reserved							ECC_EN 0x0
INITSTAT 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															INITCOMPLETEA 0x0
ERRINTEN 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTEN 0x0

Register Address Offset	Bit Fields															
ERRINTENS 0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTS 0x0
ERRINTENR 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTR 0x0
INTMODE 0x1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															INTONCMP 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							INTONOVF 0x0	Reserved							INTMODE 0x0
INTSTAT 0x20	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							DERRPENA 0x0	Reserved							SERRPENA 0x0
INTTEST 0x24	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							TDERRA 0x0	Reserved							TSERRA 0x0
MODSTAT 0x28	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															CMPFLGA 0x0

Register Address Offset	Bit Fields															
DERRADDRA 0x2C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				Address 0x0											
SERRADDRA 0x30	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				Address 0x0											
SERRCNTREG 0x3C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	SERRCNT 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SERRCNT 0x0															
ECC_Addrbus 0x40	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				ECC_AddrBUS 0x0											
ECC_RData0bus 0x44	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_RData1bus 0x48	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													ECC_RDataBUS 0x0		

Register Address Offset	Bit Fields															
ECC_RData2bus 0x4C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_RData3bus 0x50	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_WData0bus 0x54	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_WData1bus 0x58	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													ECC_WDataBUS 0x0		
ECC_WData2bus 0x5C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_WData3bus 0x60	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															

Register Address Offset	Bit Fields															
ECC_RDataecc0bus 0x64	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_RDataecc3BUS 0x0							Reserved	ECC_RDataecc2BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_RDataecc1BUS 0x0							Reserved	ECC_RDataecc0BUS 0x0						
ECC_RDataecc1bus 0x68	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_RDataecc7BUS 0x0							Reserved	ECC_RDataecc6BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_RDataecc5BUS 0x0							Reserved	ECC_RDataecc4BUS 0x0						
ECC_WDataecc0bus 0x6C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_WDataecc3BUS 0x0							Reserved	ECC_WDataecc2BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_WDataecc1BUS 0x0							Reserved	ECC_WDataecc0BUS 0x0						
ECC_WDataecc1bus 0x70	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_WDataecc7BUS 0x0							Reserved	ECC_WDataecc6BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_WDataecc5BUS 0x0							Reserved	ECC_WDataecc4BUS 0x0						
ECC_dbytectrl 0x74	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															DBEN 0x0
ECC_accctrl 0x78	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							RDWR 0x0	Reserved						ECCOVR 0x0	DATAOVR 0x0

Register Address Offset	Bit Fields															
ECC_startacc 0x7C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															ENBUSA 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															
ECC_wdctrl 0x80	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															WDEN_RAM 0x0
SERRLKUPA0 0x90	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	VALID RW 0x0	Reserved														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				Address RO 0x0											
ecc_emac2_rx_ecc_register-Block																
IP_REV_ID 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SIREV RO 0x0															
CTRL 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															INITA 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							CNT_RSTA 0x0	Reserved							ECC_EN 0x0

Register Address Offset	Bit Fields															
INITSTAT 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															INITCOM-PLETEA 0x0
ERRINTEN 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTEN 0x0
ERRINTENS 0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTS 0x0
ERRINTENR 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTR 0x0
INTMODE 0x1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															INTONCMP 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							INTONOVF 0x0	Reserved							INTMODE 0x0
INTSTAT 0x20	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							DERRPENA 0x0	Reserved							SERRPENA 0x0

Register Address Offset	Bit Fields															
INTTEST 0x24	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							TDERRA 0x0	Reserved							TSERRA 0x0
MODSTAT 0x28	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															CMPFLGA 0x0
DERRADDRA 0x2C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				Address 0x0											
SERRADDRA 0x30	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				Address 0x0											
SERRCNTREG 0x3C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	SERRCNT 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SERRCNT 0x0															
ECC_Addrbus 0x40	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				ECC_AddrBUS 0x0											

Register Address Offset	Bit Fields															
ECC_RData0bus 0x44	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_RData1bus 0x48	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													ECC_RDataBUS 0x0		
ECC_RData2bus 0x4C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_RData3bus 0x50	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_WData0bus 0x54	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_WData1bus 0x58	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													ECC_WDataBUS 0x0		

Register Address Offset	Bit Fields															
ECC_WData2bus 0x5C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_WData3bus 0x60	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_RDataecc0bus 0x64	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_RDataecc3BUS 0x0							Reserved	ECC_RDataecc2BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_RDataecc1BUS 0x0							Reserved	ECC_RDataecc0BUS 0x0						
ECC_RDataecc1bus 0x68	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_RDataecc7BUS 0x0							Reserved	ECC_RDataecc6BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_RDataecc5BUS 0x0							Reserved	ECC_RDataecc4BUS 0x0						
ECC_WDataecc0bus 0x6C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_WDataecc3BUS 0x0							Reserved	ECC_WDataecc2BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_WDataecc1BUS 0x0							Reserved	ECC_WDataecc0BUS 0x0						
ECC_WDataecc1bus 0x70	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_WDataecc7BUS 0x0							Reserved	ECC_WDataecc6BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_WDataecc5BUS 0x0							Reserved	ECC_WDataecc4BUS 0x0						

Register Address Offset	Bit Fields															
ECC_dbytectrl 0x74	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															DBEN 0x0
ECC_accctrl 0x78	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							RDWR 0x0	Reserved						ECCOVR 0x0	DATAOVR 0x0
ECC_startacc 0x7C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															ENBUSA 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															
ECC_wdctrl 0x80	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															WDEN_RAM 0x0
SERRLKUPA0 0x90	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	VALID RW 0x0	Reserved														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				Address RO 0x0											

IP_REV_ID

Module Instance	Base Address	Register Address
ecc_emac0_rx_ecc_registerBlock	0xFF8C0800	0xFF8C0800
ecc_emac1_rx_ecc_registerBlock	0xFF8C1000	0xFF8C1000

Module Instance	Base Address	Register Address
ecc_emac2_rx_ecc_registerBlock	0xFF8C1800	0xFF8C1800

Offset: 0x0

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SIREV RO 0x0															

IP_REV_ID Fields

Bit	Name	Description	Access	Reset
15:0	SIREV	IP Rev # These bits indicate the silicon revision number.	RO	0x0

CTRL

ECC Control Register

Module Instance	Base Address	Register Address
ecc_emac0_rx_ecc_registerBlock	0xFF8C0800	0xFF8C0808
ecc_emac1_rx_ecc_registerBlock	0xFF8C1000	0xFF8C1008
ecc_emac2_rx_ecc_registerBlock	0xFF8C1800	0xFF8C1808

Offset: 0x8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															INITA 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							CNT_RSTA 0x0	Reserved							ECC_EN 0x0

CTRL Fields

Bit	Name	Description	Access	Reset
16	INITA	Enable for the hardware memory initialization PORTA.	RW	0x0
8	CNT_RSTA	Enable to reset internal single-bit error counter A value to zero	RW	0x0
0	ECC_EN	Enable for the ECC detection and correction logic.	RW	0x0

INITSTAT

This bit is used to set the initialize the memory and ecc to a known value

Module Instance	Base Address	Register Address
ecc_emac0_rx_ecc_registerBlock	0xFF8C0800	0xFF8C080C
ecc_emac1_rx_ecc_registerBlock	0xFF8C1000	0xFF8C100C
ecc_emac2_rx_ecc_registerBlock	0xFF8C1800	0xFF8C180C

Offset: 0xC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															INITCOM-PLETEA 0x0

INITSTAT Fields

Bit	Name	Description	Access	Reset
0	INITCOMPLETEA	This bit is used to verify if the hardware memory initialization has completed PORTB.	RW	0x0

ERRINTEN

Error Interrupt enable

Module Instance	Base Address	Register Address
ecc_emac0_rx_ecc_registerBlock	0xFF8C0800	0xFF8C0810
ecc_emac1_rx_ecc_registerBlock	0xFF8C1000	0xFF8C1010
ecc_emac2_rx_ecc_registerBlock	0xFF8C1800	0xFF8C1810

Offset: 0x10

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															SERRINTEN 0x0

ERRINTEN Fields

Bit	Name	Description	Access	Reset
0	SERRINTEN	This bit is used to enable the single bit error interrupt of ECC RAM system	RW	0x0

ERRINTENS

Error Interrupt set

Module Instance	Base Address	Register Address
ecc_emac0_rx_ecc_registerBlock	0xFF8C0800	0xFF8C0814
ecc_emac1_rx_ecc_registerBlock	0xFF8C1000	0xFF8C1014
ecc_emac2_rx_ecc_registerBlock	0xFF8C1800	0xFF8C1814

Offset: 0x14

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															SERRINTS 0x0

ERRINTENS Fields

Bit	Name	Description	Access	Reset
0	SERRINTS	This bit is used to set the single-bit error interrupt bit.	RW	0x0

ERRINTENR

Error Interrupt reset.

Module Instance	Base Address	Register Address
ecc_emac0_rx_ecc_registerBlock	0xFF8C0800	0xFF8C0818
ecc_emac1_rx_ecc_registerBlock	0xFF8C1000	0xFF8C1018
ecc_emac2_rx_ecc_registerBlock	0xFF8C1800	0xFF8C1818

Offset: 0x18

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															SERRINTR 0x0

ERRINTENR Fields

Bit	Name	Description	Access	Reset
0	SERRINTR	This bit is used to reset the single-bit error interrupt bit. Reads reflect SERRINTEN. 1'b0: Writing of zero has no effect. 1'b1: By writing one, this bit will reset SERRINTEN bit to 0. This is performing a bitwise writing of this feature.	RW	0x0

INTMODE

Reads reflect SERRINTEN.

Module Instance	Base Address	Register Address
ecc_emac0_rx_ecc_registerBlock	0xFF8C0800	0xFF8C081C
ecc_emac1_rx_ecc_registerBlock	0xFF8C1000	0xFF8C101C
ecc_emac2_rx_ecc_registerBlock	0xFF8C1800	0xFF8C181C

Offset: 0x1C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															INTONCMP 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							INTONOVF 0x0	Reserved							INTMODE 0x0

INTMODE Fields

Bit	Name	Description	Access	Reset
16	INTONCMP	Enable interrupt on compare.	RW	0x0
8	INTONOVF	Enable interrupt on overflow.	RW	0x0
0	INTMODE	Interrupt mode for single-bit errors.	RW	0x0

INTSTAT

This bit is used to enable interrupt generation on SERR lookup table overflow. When all the entries in the table are valid=1 and this is bit is enabled, serr_req signal will be asserted.

Module Instance	Base Address	Register Address
ecc_emac0_rx_ecc_registerBlock	0xFF8C0800	0xFF8C0820
ecc_emac1_rx_ecc_registerBlock	0xFF8C1000	0xFF8C1020

Module Instance	Base Address	Register Address
ecc_emac2_rx_ecc_registerBlock	0xFF8C1800	0xFF8C1820

Offset: 0x20

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							DERRPENA 0x0	Reserved							SERRPENA 0x0

INTSTAT Fields

Bit	Name	Description	Access	Reset
8	DERRPENA	Double-bit error pending for PORTA.	RW	0x0
0	SERRPENA	Single-bit error pending for PORTA.	RW	0x0

INTTEST

This bits is used to test interrupt from ECC RAM to GIC

Module Instance	Base Address	Register Address
ecc_emac0_rx_ecc_registerBlock	0xFF8C0800	0xFF8C0824
ecc_emac1_rx_ecc_registerBlock	0xFF8C1000	0xFF8C1024
ecc_emac2_rx_ecc_registerBlock	0xFF8C1800	0xFF8C1824

Offset: 0x24

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							TDERRA 0x0	Reserved							TSERRA 0x0

INTTEST Fields

Bit	Name	Description	Access	Reset
8	TDERRA	Test PORTA Double-bit error.	RW	0x0
0	TSERRA	Test PORTA Single-bit error.	RW	0x0

MODSTAT

Counter feature status flag

Module Instance	Base Address	Register Address
ecc_emac0_rx_ecc_registerBlock	0xFF8C0800	0xFF8C0828
ecc_emac1_rx_ecc_registerBlock	0xFF8C1000	0xFF8C1028
ecc_emac2_rx_ecc_registerBlock	0xFF8C1800	0xFF8C1828

Offset: 0x28

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															CMPFLGA 0x0

MODSTAT Fields

Bit	Name	Description	Access	Reset
0	CMPFLGA	Port A compare status flag	RW	0x0

DERRADDRA

This register shows the address of PORTA current double-bit error. RAM size will determine the maximum number of address bits.

Module Instance	Base Address	Register Address
ecc_emac0_rx_ecc_registerBlock	0xFF8C0800	0xFF8C082C
ecc_emac1_rx_ecc_registerBlock	0xFF8C1000	0xFF8C102C
ecc_emac2_rx_ecc_registerBlock	0xFF8C1800	0xFF8C182C

Offset: 0x2C

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				Address 0x0											

DERRADDRA Fields

Bit	Name	Description	Access	Reset
11:0	Address	Recent double-bit error address.	RO	0x0

SERRADDRA

This register shows the address of PORTA current single-bit error. RAM size will determine the maximum number of address bits.

Module Instance	Base Address	Register Address
ecc_emac0_rx_ecc_registerBlock	0xFF8C0800	0xFF8C0830
ecc_emac1_rx_ecc_registerBlock	0xFF8C1000	0xFF8C1030
ecc_emac2_rx_ecc_registerBlock	0xFF8C1800	0xFF8C1830

Offset: 0x30

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				Address 0x0											

SERRADDRA Fields

Bit	Name	Description	Access	Reset
11:0	Address	Recent single-bit error address.	RO	0x0

SERRCNTREG

Maximum counter value for single-bit error interrupt

Module Instance	Base Address	Register Address
ecc_emac0_rx_ecc_registerBlock	0xFF8C0800	0xFF8C083C
ecc_emac1_rx_ecc_registerBlock	0xFF8C1000	0xFF8C103C
ecc_emac2_rx_ecc_registerBlock	0xFF8C1800	0xFF8C183C

Offset: 0x3C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SERRCNT 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SERRCNT 0x0															

SERRCNTREG Fields

Bit	Name	Description	Access	Reset
31:0	SERRCNT	Counter value	RW	0x0

ECC_Addrbus

MSB bit of address is determined by ADR.

Module Instance	Base Address	Register Address
ecc_emac0_rx_ecc_registerBlock	0xFF8C0800	0xFF8C0840
ecc_emac1_rx_ecc_registerBlock	0xFF8C1000	0xFF8C1040
ecc_emac2_rx_ecc_registerBlock	0xFF8C1800	0xFF8C1840

Offset: 0x40

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				ECC_AddrBUS 0x0											

ECC_Addrbus Fields

Bit	Name	Description	Access	Reset
11:0	ECC_AddrBUS	Address will be driven to RAM to either read or write the data. Address will be latched by the RAM when the Enbus is asserted.	RW	0x0

ECC_RData0bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_emac0_rx_ecc_registerBlock	0xFF8C0800	0xFF8C0844
ecc_emac1_rx_ecc_registerBlock	0xFF8C1000	0xFF8C1044
ecc_emac2_rx_ecc_registerBlock	0xFF8C1800	0xFF8C1844

Offset: 0x44

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData0bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_RDataBUS	ECC_RDataBUS[31:0].	RO	0x0

ECC_RData1bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_emac0_rx_ecc_registerBlock	0xFF8C0800	0xFF8C0848
ecc_emac1_rx_ecc_registerBlock	0xFF8C1000	0xFF8C1048
ecc_emac2_rx_ecc_registerBlock	0xFF8C1800	0xFF8C1848

Offset: 0x48

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													ECC_RDataBUS 0x0		

ECC_RData1bus Fields

Bit	Name	Description	Access	Reset
2:0	ECC_RDataBUS	ECC_RDataBUS[63:32].	RO	0x0

ECC_RData2bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_emac0_rx_ecc_registerBlock	0xFF8C0800	0xFF8C084C
ecc_emac1_rx_ecc_registerBlock	0xFF8C1000	0xFF8C104C
ecc_emac2_rx_ecc_registerBlock	0xFF8C1800	0xFF8C184C

Offset: 0x4C

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData2bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_RDataBUS	ECC_RDataBUS[95:64].	RO	0x0

ECC_RData3bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_emac0_rx_ecc_registerBlock	0xFF8C0800	0xFF8C0850
ecc_emac1_rx_ecc_registerBlock	0xFF8C1000	0xFF8C1050
ecc_emac2_rx_ecc_registerBlock	0xFF8C1800	0xFF8C1850

Offset: 0x50

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData3bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_RDataBUS	ECC_RDataBUS[127-96].	RO	0x0

ECC_WData0bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_emac0_rx_ecc_registerBlock	0xFF8C0800	0xFF8C0854
ecc_emac1_rx_ecc_registerBlock	0xFF8C1000	0xFF8C1054
ecc_emac2_rx_ecc_registerBlock	0xFF8C1800	0xFF8C1854

Offset: 0x54

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData0bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_WDataBUS	ECC_WDataBUS[31:0].	WO	0x0

ECC_WData1bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_emac0_rx_ecc_registerBlock	0xFF8C0800	0xFF8C0858

Module Instance	Base Address	Register Address
ecc_emac1_rx_ecc_registerBlock	0xFF8C1000	0xFF8C1058
ecc_emac2_rx_ecc_registerBlock	0xFF8C1800	0xFF8C1858

Offset: 0x58

Access: WO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													ECC_WDataBUS 0x0		

ECC_WData1bus Fields

Bit	Name	Description	Access	Reset
2:0	ECC_WDataBUS	ECC_WDataBUS[63:32].	WO	0x0

ECC_WData2bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_emac0_rx_ecc_registerBlock	0xFF8C0800	0xFF8C085C
ecc_emac1_rx_ecc_registerBlock	0xFF8C1000	0xFF8C105C
ecc_emac2_rx_ecc_registerBlock	0xFF8C1800	0xFF8C185C

Offset: 0x5C

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData2bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_WDataBUS	ECC_WDataBUS[95-64].	WO	0x0

ECC_WData3bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_emac0_rx_ecc_registerBlock	0xFF8C0800	0xFF8C0860
ecc_emac1_rx_ecc_registerBlock	0xFF8C1000	0xFF8C1060
ecc_emac2_rx_ecc_registerBlock	0xFF8C1800	0xFF8C1860

Offset: 0x60

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData3bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_WDataBUS	ECC_WDataBUS[127-96].	WO	0x0

ECC_RDataecc0bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_emac0_rx_ecc_registerBlock	0xFF8C0800	0xFF8C0864
ecc_emac1_rx_ecc_registerBlock	0xFF8C1000	0xFF8C1064
ecc_emac2_rx_ecc_registerBlock	0xFF8C1800	0xFF8C1864

Offset: 0x64

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	ECC_RDataecc3BUS 0x0							Reserved	ECC_RDataecc2BUS 0x0						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	ECC_RDataecc1BUS 0x0							Reserved	ECC_RDataecc0BUS 0x0						

ECC_RDataecc0bus Fields

Bit	Name	Description	Access	Reset
30:24	ECC_RDataecc3BUS	Eccdata will be read to this register field.	RO	0x0
22:16	ECC_RDataecc2BUS	Eccdata will be read to this register field.	RO	0x0

Bit	Name	Description	Access	Reset
14:8	ECC_RDataecc1BUS	Eccdata will be read to this register field.	RO	0x0
6:0	ECC_RDataecc0BUS	Eccdata will be read to this register field.	RO	0x0

ECC_RDataecc1bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_emac0_rx_ecc_registerBlock	0xFF8C0800	0xFF8C0868
ecc_emac1_rx_ecc_registerBlock	0xFF8C1000	0xFF8C1068
ecc_emac2_rx_ecc_registerBlock	0xFF8C1800	0xFF8C1868

Offset: 0x68

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	ECC_RDataecc7BUS 0x0							Reserved	ECC_RDataecc6BUS 0x0						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	ECC_RDataecc5BUS 0x0							Reserved	ECC_RDataecc4BUS 0x0						

ECC_RDataecc1bus Fields

Bit	Name	Description	Access	Reset
30:24	ECC_RDataecc7BUS	Eccdata will be read to this register field.	RO	0x0

Bit	Name	Description	Access	Reset
22:16	ECC_RDataecc6BUS	Eccdata will be read to this register field.	RO	0x0
14:8	ECC_RDataecc5BUS	Eccdata will be read to this register field.	RO	0x0
6:0	ECC_RDataecc4BUS	Eccdata will be read to this register field.	RO	0x0

ECC_WDataecc0bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_emac0_rx_ecc_registerBlock	0xFF8C0800	0xFF8C086C
ecc_emac1_rx_ecc_registerBlock	0xFF8C1000	0xFF8C106C
ecc_emac2_rx_ecc_registerBlock	0xFF8C1800	0xFF8C186C

Offset: 0x6C

Access: WO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	ECC_WDataecc3BUS 0x0							Reserved	ECC_WDataecc2BUS 0x0						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	ECC_WDataecc1BUS 0x0							Reserved	ECC_WDataecc0BUS 0x0						

ECC_WDataecc0bus Fields

Bit	Name	Description	Access	Reset
30:24	ECC_WDataecc3BUS	Eccdata from the register will be written to the RAM.	WO	0x0
22:16	ECC_WDataecc2BUS	Eccdata from the register will be written to the RAM.	WO	0x0
14:8	ECC_WDataecc1BUS	Eccdata from the register will be written to the RAM.	WO	0x0
6:0	ECC_WDataecc0BUS	Eccdata from the register will be written to the RAM.	WO	0x0

ECC_WDataecc1bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_emac0_rx_ecc_registerBlock	0xFF8C0800	0xFF8C0870
ecc_emac1_rx_ecc_registerBlock	0xFF8C1000	0xFF8C1070
ecc_emac2_rx_ecc_registerBlock	0xFF8C1800	0xFF8C1870

Offset: 0x70

Access: WO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	ECC_WDataecc7BUS 0x0							Reserved	ECC_WDataecc6BUS 0x0						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	ECC_WDataecc5BUS 0x0							Reserved	ECC_WDataecc4BUS 0x0						

ECC_WDataecc1bus Fields

Bit	Name	Description	Access	Reset
30:24	ECC_WDataecc7BUS	Eccdata from the register will be written to the RAM.	WO	0x0
22:16	ECC_WDataecc6BUS	Eccdata from the register will be written to the RAM.	WO	0x0
14:8	ECC_WDataecc5BUS	Eccdata from the register will be written to the RAM.	WO	0x0
6:0	ECC_WDataecc4BUS	Eccdata from the register will be written to the RAM.	WO	0x0

ECC_dbytectrl

ECC Data Byte Control. The maximum number of implemented byte enables equals the data width of the RAM divided by 8.

Module Instance	Base Address	Register Address
ecc_emac0_rx_ecc_registerBlock	0xFF8C0800	0xFF8C0874
ecc_emac1_rx_ecc_registerBlock	0xFF8C1000	0xFF8C1074
ecc_emac2_rx_ecc_registerBlock	0xFF8C1800	0xFF8C1874

Offset: 0x74

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															DBEN 0x0

ECC_dbytectrl Fields

Bit	Name	Description	Access	Reset
0	DBEN	Byte or word enable for access.	RW	0x0

ECC_accctrl

These bits determine which byte of data/ecc to write to RAM.

Module Instance	Base Address	Register Address
ecc_emac0_rx_ecc_registerBlock	0xFF8C0800	0xFF8C0878
ecc_emac1_rx_ecc_registerBlock	0xFF8C1000	0xFF8C1078
ecc_emac2_rx_ecc_registerBlock	0xFF8C1800	0xFF8C1878

Offset: 0x78

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							RDWR 0x0	Reserved						ECCOVR 0x0	DATAOVR 0x0

ECC_accctrl Fields

Bit	Name	Description	Access	Reset
8	RDWR	Control for read/write.	RW	0x0
1	ECCOVR	ECC Data Override.	RW	0x0

Bit	Name	Description	Access	Reset
0	DATAOVR	RAM Data Override. Override the ECC_dataBUS register with RAM data in read mode set by ECC_RW. 1'b0: Data override disabled. 1'b1: Data override enabled.	RW	0x0

ECC_startacc

These bits determine which byte of data/ecc to write to RAM.

Module Instance	Base Address	Register Address
ecc_emac0_rx_ecc_registerBlock	0xFF8C0800	0xFF8C087C
ecc_emac1_rx_ecc_registerBlock	0xFF8C1000	0xFF8C107C
ecc_emac2_rx_ecc_registerBlock	0xFF8C1800	0xFF8C187C

Offset: 0x7C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															ENBUSA 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															

ECC_startacc Fields

Bit	Name	Description	Access	Reset
16	ENBUSA	Start RAM access for PORTA.	RW	0x0

ECC_wdctrl

Bits to Enable/Disable Watch Dog Timer

Module Instance	Base Address	Register Address
ecc_emac0_rx_ecc_registerBlock	0xFF8C0800	0xFF8C0880
ecc_emac1_rx_ecc_registerBlock	0xFF8C1000	0xFF8C1080
ecc_emac2_rx_ecc_registerBlock	0xFF8C1800	0xFF8C1880

Offset: 0x80

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															WDEN_RAM 0x0

ECC_wdctrl Fields

Bit	Name	Description	Access	Reset
0	WDEN_RAM	Enable watchdog timeout for OCP register access to IP RAM.	RW	0x0

SERRLKUPA0

Single-bit error address in LOOKUP TABLE for PORTA.

Module Instance	Base Address	Register Address
ecc_emac0_rx_ecc_registerBlock	0xFF8C0800	0xFF8C0890
ecc_emac1_rx_ecc_registerBlock	0xFF8C1000	0xFF8C1090
ecc_emac2_rx_ecc_registerBlock	0xFF8C1800	0xFF8C1890

Offset: 0x90

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VALID RW 0x0	Reserved														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				Address RO 0x0											

SERRLKUPA0 Fields

Bit	Name	Description	Access	Reset
31	VALID	Valid flag bit. Valid bit indicates if the address in this register is current or stale.	RW	0x0
11:0	Address	Recent Single-bit error address. This register shows the address of the each single-bit error. RAM size will determine the maximum number of address bits. If ram size is 32 Kbytes, bit 30-16 will be reserved and read as zero.	RO	0x0

emac_tx_ecc Address Map

Module Instance	Base Address	End Address
ecc_emac0_tx_ecc_registerBlock	0xFF8C0C00	0xFF8C0FFF
ecc_emac1_tx_ecc_registerBlock	0xFF8C1400	0xFF8C17FF
ecc_emac2_tx_ecc_registerBlock	0xFF8C1C00	0xFF8C1FFF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
IP_REV_ID on page 11-82	0x0	32	RO	0x0	
CTRL on page 11-83	0x8	32	RW	0x0	ECC Control Register

Register	Offset	Width	Access	Reset Value	Description
INITSTAT on page 11-84	0xC	32	RW	0x0	This bit is used to set the initialize the memory and ecc to a known value
ERRINTEN on page 11-85	0x10	32	RW	0x0	Error Interrupt enable
ERRINTENS on page 11-86	0x14	32	RW	0x0	Error Interrupt set
ERRINTENR on page 11-87	0x18	32	RW	0x0	Error Interrupt reset.
INTMODE on page 11-87	0x1C	32	RW	0x0	Reads reflect SERRINTEN.
INTTEST on page 11-88	0x24	32	RW	0x0	This bits is used to test interrupt from ECC RAM to GIC
MODSTAT on page 11-89	0x28	32	RW	0x0	Counter feature status flag
DERRADDRA on page 11-90	0x2C	32	RO	0x0	This register shows the address of PORTA current double-bit error. RAM size will determine the maximum number of address bits.
SERRADDRA on page 11-91	0x30	32	RO	0x0	This register shows the address of PORTA current single-bit error. RAM size will determine the maximum number of address bits.
SERRCNTREG on page 11-91	0x3C	32	RW	0x0	Maximum counter value for single-bit error interrupt
ECC_Addrbus on page 11-92	0x40	32	RW	0x0	MSB bit of address is determined by ADR.
ECC_RData0bus on page 11-93	0x44	32	RO	0x0	Data will be read to this register field.
ECC_RData1bus on page 11-94	0x48	32	RO	0x0	Data will be read to this register field.

Register	Offset	Width	Access	Reset Value	Description
ECC_RData2bus on page 11-94	0x4C	32	RO	0x0	Data will be read to this register field.
ECC_RData3bus on page 11-95	0x50	32	RO	0x0	Data will be read to this register field.
ECC_WData0bus on page 11-96	0x54	32	WO	0x0	Data from the register will be written to the RAM.
ECC_WData1bus on page 11-97	0x58	32	WO	0x0	Data from the register will be written to the RAM.
ECC_WData2bus on page 11-97	0x5C	32	WO	0x0	Data from the register will be written to the RAM.
ECC_WData3bus on page 11-98	0x60	32	WO	0x0	Data from the register will be written to the RAM.
ECC_RDataecc0bus on page 11-99	0x64	32	RO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.
ECC_RDataecc1bus on page 11-100	0x68	32	RO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.
ECC_WDataecc0bus on page 11-101	0x6C	32	WO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Register	Offset	Width	Access	Reset Value	Description
ECC_WDataecc1bus on page 11-102	0x70	32	WO	0x0	The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.
ECC_dbytectrl on page 11-103	0x74	32	RW	0x0	Max number of implemented byte enabled is DAT/8
ECC_accctrl on page 11-104	0x78	32	RW	0x0	These bits determine which byte of data/ecc to write to RAM.
ECC_startacc on page 11-105	0x7C	32	RW	0x0	These bits determine which byte of data/ecc to write to RAM.
ECC_wdctrl on page 11-105	0x80	32	RW	0x0	Bits to Enable/Disable Watch Dog Timer
SERRLKUPA0 on page 11-106	0x90	32	RW	0x0	Single-bit error address in LOOKUP TABLE for PORTA.
INTSTAT on page 11-107	0x	32	RW	0x0	This bit is used to enable interrupt generation on SERR lookup table overflow. When all the entries in the table are valid=1 and this is bit is enabled, serr_req signal will be asserted.

emac_tx_ecc Summary

Module Instance	Base Address
ecc_emac0_tx_ecc_registerBlock	0xFF8C0C00
ecc_emac1_tx_ecc_registerBlock	0xFF8C1400
ecc_emac2_tx_ecc_registerBlock	0xFF8C1C00

Register Address Offset	Bit Fields															
ecc_emac0_tx_ecc_register-Block																
IP_REV_ID 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SIREV RO 0x0															
CTRL 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															INITA 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							CNT_RSTA 0x0	Reserved							ECC_EN 0x0
INITSTAT 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															INITCOMPLETEA 0x0
ERRINTEN 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTEN 0x0
ERRINTENS 0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTS 0x0
ERRINTENR 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTR 0x0

Register Address Offset	Bit Fields															
INTMODE 0x1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															INTONCMP 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							INTONOVF 0x0	Reserved							INTMODE 0x0
INTTEST 0x24	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							TDERRA 0x0	Reserved							TSERRA 0x0
MODSTAT 0x28	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															CMPFLGA 0x0
DERRADDRA 0x2C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						Address 0x0									
SERRADDRA 0x30	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						Address 0x0									
SERRCNTREG 0x3C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	SERRCNT 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SERRCNT 0x0															

Register Address Offset	Bit Fields															
ECC_Addrbus 0x40	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						ECC_AddrBUS 0x0									
ECC_RData0bus 0x44	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_RData1bus 0x48	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													ECC_RDataBUS 0x0		
ECC_RData2bus 0x4C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_RData3bus 0x50	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_WData0bus 0x54	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															

Register Address Offset	Bit Fields															
ECC_WData1bus 0x58	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													ECC_WDataBUS 0x0		
ECC_WData2bus 0x5C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_WData3bus 0x60	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_RDataecc0bus 0x64	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_RDataecc3BUS 0x0							Reserved	ECC_RDataecc2BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_RDataecc1BUS 0x0							Reserved	ECC_RDataecc0BUS 0x0						
ECC_RDataecc1bus 0x68	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_RDataecc7BUS 0x0							Reserved	ECC_RDataecc6BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_RDataecc5BUS 0x0							Reserved	ECC_RDataecc4BUS 0x0						
ECC_WDataecc0bus 0x6C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_WDataecc3BUS 0x0							Reserved	ECC_WDataecc2BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_WDataecc1BUS 0x0							Reserved	ECC_WDataecc0BUS 0x0						

Register Address Offset	Bit Fields															
ECC_WDataecc1bus 0x70	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_WDataecc7BUS 0x0							Reserved	ECC_WDataecc6BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_WDataecc5BUS 0x0							Reserved	ECC_WDataecc4BUS 0x0						
ECC_dbytectrl 0x74	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															DBEN 0x0
ECC_accctrl 0x78	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							RDWR 0x0	Reserved						ECCOVR 0x0	DATAOVR 0x0
ECC_startacc 0x7C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															ENBUSA 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															
ECC_wdctrl 0x80	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															WDEN_RAM 0x0
SERRLKUPA0 0x90	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	VALID RW 0x0	Reserved														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						Address RO 0x0									

Register Address Offset	Bit Fields															
INTSTAT 0x32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							DERRPENA 0x0	Reserved							SERRPENA 0x0
ecc_emac1_tx_ecc_register-Block																
IP_REV_ID 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SIREV RO 0x0															
CTRL 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															INITA 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							CNT_RSTA 0x0	Reserved							ECC_EN 0x0
INITSTAT 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															INITCOM-PLETEA 0x0
ERRINTEN 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTEN 0x0
ERRINTENS 0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTS 0x0

Register Address Offset	Bit Fields															
ERRINTENR 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTR 0x0
INTMODE 0x1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															INTONCMP 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							INTONOVF 0x0	Reserved							INTMODE 0x0
INTTEST 0x24	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							TDERRA 0x0	Reserved							TSERRA 0x0
MODSTAT 0x28	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															CMPFLGA 0x0
DERRADDRA 0x2C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						Address 0x0									
SERRADDRA 0x30	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						Address 0x0									

Register Address Offset	Bit Fields															
SERRCNTREG 0x3C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	SERRCNT 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SERRCNT 0x0															
ECC_Addrbus 0x40	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						ECC_AddrBUS 0x0									
ECC_RData0bus 0x44	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_RData1bus 0x48	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													ECC_RDataBUS 0x0		
ECC_RData2bus 0x4C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_RData3bus 0x50	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															

Register Address Offset	Bit Fields															
ECC_WData0bus 0x54	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_WData1bus 0x58	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													ECC_WDataBUS 0x0		
ECC_WData2bus 0x5C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_WData3bus 0x60	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_RDataecc0bus 0x64	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_RDataecc3BUS 0x0							Reserved	ECC_RDataecc2BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_RDataecc1BUS 0x0							Reserved	ECC_RDataecc0BUS 0x0						
ECC_RDataecc1bus 0x68	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_RDataecc7BUS 0x0							Reserved	ECC_RDataecc6BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_RDataecc5BUS 0x0							Reserved	ECC_RDataecc4BUS 0x0						

Register Address Offset	Bit Fields															
ECC_WDataecc0bus 0x6C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_WDataecc3BUS 0x0							Reserved	ECC_WDataecc2BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_WDataecc1BUS 0x0							Reserved	ECC_WDataecc0BUS 0x0						
ECC_WDataecc1bus 0x70	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_WDataecc7BUS 0x0							Reserved	ECC_WDataecc6BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_WDataecc5BUS 0x0							Reserved	ECC_WDataecc4BUS 0x0						
ECC_dbytectrl 0x74	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															DBEN 0x0
ECC_accctrl 0x78	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							RDWR 0x0	Reserved						ECCOVR 0x0	DATAOVR 0x0
ECC_startacc 0x7C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															ENBUSA 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															
ECC_wdctrl 0x80	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															WDEN_RAM 0x0

Register Address Offset	Bit Fields															
SERRLKUPA0 0x90	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	VALID RW 0x0	Reserved														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						Address RO 0x0									
INTSTAT 0x32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							DERRPENA 0x0	Reserved							SERRPENA 0x0
ecc_emac2_tx_ecc_register-Block																
IP_REV_ID 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SIREV RO 0x0															
CTRL 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															INITA 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							CNT_RSTA 0x0	Reserved							ECC_EN 0x0
INITSTAT 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															INITCOM-PLETEA 0x0

Register Address Offset	Bit Fields															
ERRINTEN 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTEN 0x0
ERRINTENS 0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTS 0x0
ERRINTENR 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SERRINTR 0x0
INTMODE 0x1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															INTONCMP 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							INTONOVF 0x0	Reserved							INTMODE 0x0
INTTEST 0x24	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							TDERRA 0x0	Reserved							TSERRA 0x0
MODSTAT 0x28	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															CMPFLGA 0x0

Register Address Offset	Bit Fields															
DERRADDRA 0x2C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						Address 0x0									
SERRADDRA 0x30	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						Address 0x0									
SERRCNTREG 0x3C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	SERRCNT 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SERRCNT 0x0															
ECC_Addrbus 0x40	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						ECC_AddrBUS 0x0									
ECC_RData0bus 0x44	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_RData1bus 0x48	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													ECC_RDataBUS 0x0		

Register Address Offset	Bit Fields															
ECC_RData2bus 0x4C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_RData3bus 0x50	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_RDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_RDataBUS 0x0															
ECC_WData0bus 0x54	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_WData1bus 0x58	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													ECC_WDataBUS 0x0		
ECC_WData2bus 0x5C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															
ECC_WData3bus 0x60	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ECC_WDataBUS 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECC_WDataBUS 0x0															

Register Address Offset	Bit Fields															
ECC_RDataecc0bus 0x64	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_RDataecc3BUS 0x0							Reserved	ECC_RDataecc2BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_RDataecc1BUS 0x0							Reserved	ECC_RDataecc0BUS 0x0						
ECC_RDataecc1bus 0x68	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_RDataecc7BUS 0x0							Reserved	ECC_RDataecc6BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_RDataecc5BUS 0x0							Reserved	ECC_RDataecc4BUS 0x0						
ECC_WDataecc0bus 0x6C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_WDataecc3BUS 0x0							Reserved	ECC_WDataecc2BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_WDataecc1BUS 0x0							Reserved	ECC_WDataecc0BUS 0x0						
ECC_WDataecc1bus 0x70	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	ECC_WDataecc7BUS 0x0							Reserved	ECC_WDataecc6BUS 0x0						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ECC_WDataecc5BUS 0x0							Reserved	ECC_WDataecc4BUS 0x0						
ECC_dbytectrl 0x74	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															DBEN 0x0
ECC_accctrl 0x78	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							RDWR 0x0	Reserved						ECCOVR 0x0	DATAOVR 0x0

Register Address Offset	Bit Fields															
ECC_startacc 0x7C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															ENBUSA 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															
ECC_wdctrl 0x80	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															WDEN_RAM 0x0
SERRLKUPA0 0x90	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	VALID RW 0x0	Reserved														
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						Address RO 0x0									
INTSTAT 0x32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							DERRPENA 0x0	Reserved							SERRPENA 0x0

IP_REV_ID

Module Instance	Base Address	Register Address
ecc_emac0_tx_ecc_registerBlock	0xFF8C0C00	0xFF8C0C00
ecc_emac1_tx_ecc_registerBlock	0xFF8C1400	0xFF8C1400
ecc_emac2_tx_ecc_registerBlock	0xFF8C1C00	0xFF8C1C00

Offset: 0x0

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SIREV RO 0x0															

IP_REV_ID Fields

Bit	Name	Description	Access	Reset
15:0	SIREV	IP Rev # These bits indicate the silicon revision number.	RO	0x0

CTRL

ECC Control Register

Module Instance	Base Address	Register Address
ecc_emac0_tx_ecc_registerBlock	0xFF8C0C00	0xFF8C0C08
ecc_emac1_tx_ecc_registerBlock	0xFF8C1400	0xFF8C1408
ecc_emac2_tx_ecc_registerBlock	0xFF8C1C00	0xFF8C1C08

Offset: 0x8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															INITA 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							CNT_RSTA 0x0	Reserved							ECC_EN 0x0

CTRL Fields

Bit	Name	Description	Access	Reset
16	INITA	Enable for the hardware memory initialization PORTA.	RW	0x0
8	CNT_RSTA	Enable to reset internal single-bit error counter A value to zero	RW	0x0
0	ECC_EN	Enable for the ECC detection and correction logic.	RW	0x0

INITSTAT

This bit is used to set the initialize the memory and ecc to a known value

Module Instance	Base Address	Register Address
ecc_emac0_tx_ecc_registerBlock	0xFF8C0C00	0xFF8C0C0C
ecc_emac1_tx_ecc_registerBlock	0xFF8C1400	0xFF8C140C
ecc_emac2_tx_ecc_registerBlock	0xFF8C1C00	0xFF8C1C0C

Offset: 0xC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															INITCOM-PLETEA 0x0

INITSTAT Fields

Bit	Name	Description	Access	Reset
0	INITCOMPLETEA	This bit is used to verify if the hardware memory initialization has completed PORTB.	RW	0x0

ERRINTEN

Error Interrupt enable

Module Instance	Base Address	Register Address
ecc_emac0_tx_ecc_registerBlock	0xFF8C0C00	0xFF8C0C10
ecc_emac1_tx_ecc_registerBlock	0xFF8C1400	0xFF8C1410
ecc_emac2_tx_ecc_registerBlock	0xFF8C1C00	0xFF8C1C10

Offset: 0x10

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															SERRINTEN 0x0

ERRINTEN Fields

Bit	Name	Description	Access	Reset
0	SERRINTEN	This bit is used to enable the single bit error interrupt of ECC RAM system	RW	0x0

ERRINTENS

Error Interrupt set

Module Instance	Base Address	Register Address
ecc_emac0_tx_ecc_registerBlock	0xFF8C0C00	0xFF8C0C14
ecc_emac1_tx_ecc_registerBlock	0xFF8C1400	0xFF8C1414
ecc_emac2_tx_ecc_registerBlock	0xFF8C1C00	0xFF8C1C14

Offset: 0x14

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															SERRINTS 0x0

ERRINTENS Fields

Bit	Name	Description	Access	Reset
0	SERRINTS	This bit is used to set the single-bit error interrupt bit.	RW	0x0

ERRINTENR

Error Interrupt reset.

Module Instance	Base Address	Register Address
ecc_emac0_tx_ecc_registerBlock	0xFF8C0C00	0xFF8C0C18
ecc_emac1_tx_ecc_registerBlock	0xFF8C1400	0xFF8C1418
ecc_emac2_tx_ecc_registerBlock	0xFF8C1C00	0xFF8C1C18

Offset: 0x18

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															SERRINTR 0x0

ERRINTENR Fields

Bit	Name	Description	Access	Reset
0	SERRINTR	This bit is used to reset the single-bit error interrupt bit. Reads reflect SERRINTEN. 1'b0: Writing of zero has no effect. 1'b1: By writing one, this bit will reset SERRINTEN bit to 0. This is performing a bitwise writing of this feature.	RW	0x0

INTMODE

Reads reflect SERRINTEN.

Module Instance	Base Address	Register Address
ecc_emac0_tx_ecc_registerBlock	0xFF8C0C00	0xFF8C0C1C
ecc_emac1_tx_ecc_registerBlock	0xFF8C1400	0xFF8C141C
ecc_emac2_tx_ecc_registerBlock	0xFF8C1C00	0xFF8C1C1C

Offset: 0x1C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															INTONCMP 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							INTONOVF 0x0	Reserved							INTMODE 0x0

INTMODE Fields

Bit	Name	Description	Access	Reset
16	INTONCMP	Enable interrupt on compare.	RW	0x0
8	INTONOVF	Enable interrupt on overflow.	RW	0x0
0	INTMODE	Interrupt mode for single-bit errors.	RW	0x0

INTTEST

This bits is used to test interrupt from ECC RAM to GIC

Module Instance	Base Address	Register Address
ecc_emac0_tx_ecc_registerBlock	0xFF8C0C00	0xFF8C0C24
ecc_emac1_tx_ecc_registerBlock	0xFF8C1400	0xFF8C1424
ecc_emac2_tx_ecc_registerBlock	0xFF8C1C00	0xFF8C1C24

Offset: 0x24

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							TDERRA 0x0	Reserved							TSERRA 0x0

INTTEST Fields

Bit	Name	Description	Access	Reset
8	TDERRA	Test PORTA Double-bit error.	RW	0x0
0	TSERRA	Test PORTA Single-bit error.	RW	0x0

MODSTAT

Counter feature status flag

Module Instance	Base Address	Register Address
ecc_emac0_tx_ecc_registerBlock	0xFF8C0C00	0xFF8C0C28
ecc_emac1_tx_ecc_registerBlock	0xFF8C1400	0xFF8C1428
ecc_emac2_tx_ecc_registerBlock	0xFF8C1C00	0xFF8C1C28

Offset: 0x28

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															CMPFLGA 0x0

MODSTAT Fields

Bit	Name	Description	Access	Reset
0	CMPFLGA	Port A compare status flag	RW	0x0

DERRADDRA

This register shows the address of PORTA current double-bit error. RAM size will determine the maximum number of address bits.

Module Instance	Base Address	Register Address
ecc_emac0_tx_ecc_registerBlock	0xFF8C0C00	0xFF8C0C2C
ecc_emac1_tx_ecc_registerBlock	0xFF8C1400	0xFF8C142C
ecc_emac2_tx_ecc_registerBlock	0xFF8C1C00	0xFF8C1C2C

Offset: 0x2C

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						Address 0x0									

DERRADDRA Fields

Bit	Name	Description	Access	Reset
9:0	Address	Recent double-bit error address.	RO	0x0

SERRADDRA

This register shows the address of PORTA current single-bit error. RAM size will determine the maximum number of address bits.

Module Instance	Base Address	Register Address
ecc_emac0_tx_ecc_registerBlock	0xFF8C0C00	0xFF8C0C30
ecc_emac1_tx_ecc_registerBlock	0xFF8C1400	0xFF8C1430
ecc_emac2_tx_ecc_registerBlock	0xFF8C1C00	0xFF8C1C30

Offset: 0x30

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						Address 0x0									

SERRADDRA Fields

Bit	Name	Description	Access	Reset
9:0	Address	Recent single-bit error address.	RO	0x0

SERRCNTREG

Maximum counter value for single-bit error interrupt

Module Instance	Base Address	Register Address
ecc_emac0_tx_ecc_registerBlock	0xFF8C0C00	0xFF8C0C3C
ecc_emac1_tx_ecc_registerBlock	0xFF8C1400	0xFF8C143C
ecc_emac2_tx_ecc_registerBlock	0xFF8C1C00	0xFF8C1C3C

Offset: 0x3C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SERRCNT 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SERRCNT 0x0															

SERRCNTREG Fields

Bit	Name	Description	Access	Reset
31:0	SERRCNT	Counter value	RW	0x0

ECC_Addrbus

MSB bit of address is determined by ADR.

Module Instance	Base Address	Register Address
ecc_emac0_tx_ecc_registerBlock	0xFF8C0C00	0xFF8C0C40
ecc_emac1_tx_ecc_registerBlock	0xFF8C1400	0xFF8C1440
ecc_emac2_tx_ecc_registerBlock	0xFF8C1C00	0xFF8C1C40

Offset: 0x40

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						ECC_AddrBUS 0x0									

ECC_Addrbus Fields

Bit	Name	Description	Access	Reset
9:0	ECC_AddrBUS	Address will be driven to RAM to either read or write the data. Address will be latched by the RAM when the Enbus is asserted.	RW	0x0

ECC_RData0bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_emac0_tx_ecc_registerBlock	0xFF8C0C00	0xFF8C0C44
ecc_emac1_tx_ecc_registerBlock	0xFF8C1400	0xFF8C1444
ecc_emac2_tx_ecc_registerBlock	0xFF8C1C00	0xFF8C1C44

Offset: 0x44

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData0bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_RDataBUS	ECC_RDataBUS[31:0].	RO	0x0

ECC_RData1bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_emac0_tx_ecc_registerBlock	0xFF8C0C00	0xFF8C0C48
ecc_emac1_tx_ecc_registerBlock	0xFF8C1400	0xFF8C1448
ecc_emac2_tx_ecc_registerBlock	0xFF8C1C00	0xFF8C1C48

Offset: 0x48

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													ECC_RDataBUS 0x0		

ECC_RData1bus Fields

Bit	Name	Description	Access	Reset
2:0	ECC_RDataBUS	ECC_RDataBUS[63:32].	RO	0x0

ECC_RData2bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_emac0_tx_ecc_registerBlock	0xFF8C0C00	0xFF8C0C4C
ecc_emac1_tx_ecc_registerBlock	0xFF8C1400	0xFF8C144C
ecc_emac2_tx_ecc_registerBlock	0xFF8C1C00	0xFF8C1C4C

Offset: 0x4C

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData2bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_RDataBUS	ECC_RDataBUS[95:64].	RO	0x0

ECC_RData3bus

Data will be read to this register field.

Module Instance	Base Address	Register Address
ecc_emac0_tx_ecc_registerBlock	0xFF8C0C00	0xFF8C0C50
ecc_emac1_tx_ecc_registerBlock	0xFF8C1400	0xFF8C1450
ecc_emac2_tx_ecc_registerBlock	0xFF8C1C00	0xFF8C1C50

Offset: 0x50

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_RDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_RDataBUS 0x0															

ECC_RData3bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_RDataBUS	ECC_RDataBUS[127-96].	RO	0x0

ECC_WData0bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_emac0_tx_ecc_registerBlock	0xFF8C0C00	0xFF8C0C54
ecc_emac1_tx_ecc_registerBlock	0xFF8C1400	0xFF8C1454
ecc_emac2_tx_ecc_registerBlock	0xFF8C1C00	0xFF8C1C54

Offset: 0x54

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData0bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_WDataBUS	ECC_WDataBUS[31:0].	WO	0x0

ECC_WData1bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_emac0_tx_ecc_registerBlock	0xFF8C0C00	0xFF8C0C58
ecc_emac1_tx_ecc_registerBlock	0xFF8C1400	0xFF8C1458
ecc_emac2_tx_ecc_registerBlock	0xFF8C1C00	0xFF8C1C58

Offset: 0x58

Access: WO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													ECC_WDataBUS 0x0		

ECC_WData1bus Fields

Bit	Name	Description	Access	Reset
2:0	ECC_WDataBUS	ECC_WDataBUS[63:32].	WO	0x0

ECC_WData2bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_emac0_tx_ecc_registerBlock	0xFF8C0C00	0xFF8C0C5C
ecc_emac1_tx_ecc_registerBlock	0xFF8C1400	0xFF8C145C
ecc_emac2_tx_ecc_registerBlock	0xFF8C1C00	0xFF8C1C5C

Offset: 0x5C

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData2bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_WDataBUS	ECC_WDataBUS[95-64].	WO	0x0

ECC_WData3bus

Data from the register will be written to the RAM.

Module Instance	Base Address	Register Address
ecc_emac0_tx_ecc_registerBlock	0xFF8C0C00	0xFF8C0C60
ecc_emac1_tx_ecc_registerBlock	0xFF8C1400	0xFF8C1460
ecc_emac2_tx_ecc_registerBlock	0xFF8C1C00	0xFF8C1C60

Offset: 0x60

Access: WO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECC_WDataBUS 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_WDataBUS 0x0															

ECC_WData3bus Fields

Bit	Name	Description	Access	Reset
31:0	ECC_WDataBUS	ECC_WDataBUS[127-96].	WO	0x0

ECC_RDataecc0bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_emac0_tx_ecc_registerBlock	0xFF8C0C00	0xFF8C0C64
ecc_emac1_tx_ecc_registerBlock	0xFF8C1400	0xFF8C1464
ecc_emac2_tx_ecc_registerBlock	0xFF8C1C00	0xFF8C1C64

Offset: 0x64

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	ECC_RDataecc3BUS 0x0							Reserved	ECC_RDataecc2BUS 0x0						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	ECC_RDataecc1BUS 0x0							Reserved	ECC_RDataecc0BUS 0x0						

ECC_RDataecc0bus Fields

Bit	Name	Description	Access	Reset
30:24	ECC_RDataecc3BUS	Eccdata will be read to this register field.	RO	0x0
22:16	ECC_RDataecc2BUS	Eccdata will be read to this register field.	RO	0x0
14:8	ECC_RDataecc1BUS	Eccdata will be read to this register field.	RO	0x0
6:0	ECC_RDataecc0BUS	Eccdata will be read to this register field.	RO	0x0

ECC_RDataecc1bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_emac0_tx_ecc_registerBlock	0xFF8C0C00	0xFF8C0C68
ecc_emac1_tx_ecc_registerBlock	0xFF8C1400	0xFF8C1468
ecc_emac2_tx_ecc_registerBlock	0xFF8C1C00	0xFF8C1C68

Offset: 0x68

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	ECC_RDataecc7BUS 0x0							Reserved	ECC_RDataecc6BUS 0x0						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	ECC_RDataecc5BUS 0x0							Reserved	ECC_RDataecc4BUS 0x0						

ECC_RDataecc1bus Fields

Bit	Name	Description	Access	Reset
30:24	ECC_RDataecc7BUS	Eccdata will be read to this register field.	RO	0x0
22:16	ECC_RDataecc6BUS	Eccdata will be read to this register field.	RO	0x0
14:8	ECC_RDataecc5BUS	Eccdata will be read to this register field.	RO	0x0
6:0	ECC_RDataecc4BUS	Eccdata will be read to this register field.	RO	0x0

ECC_WDataecc0bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_emac0_tx_ecc_registerBlock	0xFF8C0C00	0xFF8C0C6C
ecc_emac1_tx_ecc_registerBlock	0xFF8C1400	0xFF8C146C
ecc_emac2_tx_ecc_registerBlock	0xFF8C1C00	0xFF8C1C6C

Offset: 0x6C

Access: WO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	ECC_WDataecc3BUS 0x0							Reserved	ECC_WDataecc2BUS 0x0						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	ECC_WDataecc1BUS 0x0							Reserved	ECC_WDataecc0BUS 0x0						

ECC_WDataecc0bus Fields

Bit	Name	Description	Access	Reset
30:24	ECC_WDataecc3BUS	Eccdata from the register will be written to the RAM.	WO	0x0
22:16	ECC_WDataecc2BUS	Eccdata from the register will be written to the RAM.	WO	0x0
14:8	ECC_WDataecc1BUS	Eccdata from the register will be written to the RAM.	WO	0x0
6:0	ECC_WDataecc0BUS	Eccdata from the register will be written to the RAM.	WO	0x0

ECC_WDataecc1bus

The msb bit for the register is configured based on DAT parameter (RAM word size). Unimplemented bytes of this register will be reserved.

Module Instance	Base Address	Register Address
ecc_emac0_tx_ecc_registerBlock	0xFF8C0C00	0xFF8C0C70
ecc_emac1_tx_ecc_registerBlock	0xFF8C1400	0xFF8C1470
ecc_emac2_tx_ecc_registerBlock	0xFF8C1C00	0xFF8C1C70

Offset: 0x70

Access: WO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	ECC_WDataecc7BUS 0x0							Reserved	ECC_WDataecc6BUS 0x0						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	ECC_WDataecc5BUS 0x0							Reserved	ECC_WDataecc4BUS 0x0						

ECC_WDataecc1bus Fields

Bit	Name	Description	Access	Reset
30:24	ECC_WDataecc7BUS	Eccdata from the register will be written to the RAM.	WO	0x0
22:16	ECC_WDataecc6BUS	Eccdata from the register will be written to the RAM.	WO	0x0
14:8	ECC_WDataecc5BUS	Eccdata from the register will be written to the RAM.	WO	0x0
6:0	ECC_WDataecc4BUS	Eccdata from the register will be written to the RAM.	WO	0x0

ECC_dbytectrl

ECC Data Byte Control. The maximum number of implemented byte enables equals the data width of the RAM divided by 8.

Module Instance	Base Address	Register Address
ecc_emac0_tx_ecc_registerBlock	0xFF8C0C00	0xFF8C0C74
ecc_emac1_tx_ecc_registerBlock	0xFF8C1400	0xFF8C1474
ecc_emac2_tx_ecc_registerBlock	0xFF8C1C00	0xFF8C1C74

Offset: 0x74

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															DBEN 0x0

ECC_dbytectrl Fields

Bit	Name	Description	Access	Reset
0	DBEN	Byte or word enable for access.	RW	0x0

ECC_accctrl

These bits determine which byte of data/ecc to write to RAM.

Module Instance	Base Address	Register Address
ecc_emac0_tx_ecc_registerBlock	0xFF8C0C00	0xFF8C0C78
ecc_emac1_tx_ecc_registerBlock	0xFF8C1400	0xFF8C1478
ecc_emac2_tx_ecc_registerBlock	0xFF8C1C00	0xFF8C1C78

Offset: 0x78

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							RDWR 0x0	Reserved						ECCOVR 0x0	DATAOVR 0x0

ECC_accctrl Fields

Bit	Name	Description	Access	Reset
8	RDWR	Control for read/write.	RW	0x0
1	ECCOVR	ECC Data Override.	RW	0x0

Bit	Name	Description	Access	Reset
0	DATAOVR	RAM Data Override. Override the ECC_dataBUS register with RAM data in read mode set by ECC_RW. 1'b0: Data override disabled. 1'b1: Data override enabled.	RW	0x0

ECC_startacc

These bits determine which byte of data/ecc to write to RAM.

Module Instance	Base Address	Register Address
ecc_emac0_tx_ecc_registerBlock	0xFF8C0C00	0xFF8C0C7C
ecc_emac1_tx_ecc_registerBlock	0xFF8C1400	0xFF8C147C
ecc_emac2_tx_ecc_registerBlock	0xFF8C1C00	0xFF8C1C7C

Offset: 0x7C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															ENBUSA 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															

ECC_startacc Fields

Bit	Name	Description	Access	Reset
16	ENBUSA	Start RAM access for PORTA.	RW	0x0

ECC_wdctrl

Bits to Enable/Disable Watch Dog Timer

Module Instance	Base Address	Register Address
ecc_emac0_tx_ecc_registerBlock	0xFF8C0C00	0xFF8C0C80
ecc_emac1_tx_ecc_registerBlock	0xFF8C1400	0xFF8C1480
ecc_emac2_tx_ecc_registerBlock	0xFF8C1C00	0xFF8C1C80

Offset: 0x80

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															WDEN_RAM 0x0

ECC_wdctrl Fields

Bit	Name	Description	Access	Reset
0	WDEN_RAM	Enable watchdog timeout for OCP register access to IP RAM.	RW	0x0

SERRLKUPA0

Single-bit error address in LOOKUP TABLE for PORTA.

Module Instance	Base Address	Register Address
ecc_emac0_tx_ecc_registerBlock	0xFF8C0C00	0xFF8C0C90
ecc_emac1_tx_ecc_registerBlock	0xFF8C1400	0xFF8C1490
ecc_emac2_tx_ecc_registerBlock	0xFF8C1C00	0xFF8C1C90

Offset: 0x90

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VALID RW 0x0	Reserved														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						Address RO 0x0									

SERRLKUPA0 Fields

Bit	Name	Description	Access	Reset
31	VALID	Valid flag bit. Valid bit indicates if the address in this register is current or stale.	RW	0x0
9:0	Address	Recent Single-bit error address. This register shows the address of the each single-bit error. RAM size will determine the maximum number of address bits. If ram size is 32 Kbytes, bit 30-16 will be reserved and read as zero.	RO	0x0

INTSTAT

This bit is used to enable interrupt generation on SERR lookup table overflow. When all the entries in the table are valid=1 and this is bit is enabled, serr_req signal will be asserted.

Module Instance	Base Address	Register Address
ecc_emac0_tx_ecc_registerBlock	0xFF8C0C00	0xFF8C0C32
ecc_emac1_tx_ecc_registerBlock	0xFF8C1400	0xFF8C1432
ecc_emac2_tx_ecc_registerBlock	0xFF8C1C00	0xFF8C1C32

Offset: 0x32

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							DERRPENA 0x0	Reserved							SERRPENA 0x0

INTSTAT Fields

Bit	Name	Description	Access	Reset
8	DERRPENA	Double-bit error pending for PORTA.	RW	0x0
0	SERRPENA	Single-bit error pending for PORTA.	RW	0x0

Document Revision History

Table 17-29: Document Revision History

Date	Version	Changes
October 2016	2016.10.28	• Bit 16 updated **Table 17-6** • Updated **Figure 17-15** • Added **Figure 17-16**
May 2016	2016.05.27	Removed references to the Application Interface (also known as the switch interface). This feature is not supported in this device.
May 2016	2016.05.03	Maintenance release.

November 2015	2015.11.02	• Added emac_phy_mac_speed_o signals to "FPGA EMAC I/O Signals" section
		• Added the following subsections in the "Layer 3 and Layer 4 Filters" section:
		  • Layer 3 and Layer 4 Filters Register Set
		  • Layer 3 Filtering
		  • Layer 4 Filtering
		• Added internal clock diagram to "Clock Structure" section

Date	Version	Changes
May 2015	2015.05.04	• Corrected IEEE 1588 timestamp resolution in the "EMAC Block Diagram and System Integration" section and the "IEEE 1588-2002 Timestamps" section • Added clarification for `phy_txclk_o` and `phy_clk_rx_i` in the "HPS EMAC I/O Signals" • Added subsections "Ordinary Clock," "Boundary Clock," "End-to-End Transparent Clock" and "Peer-to-Peer Transparent Clock" in the "Clock Type" section • Added clarification in the "EMAC Module Clock Inputs and Outputs" table for the description of `phy_clk_rx_i` in the "Clock Structure" section • Added "EMAC ECC RAM Reset" subsection in "Taking the Ethernet Out of Reset" section • Added address map and register definitions
December 2014	2014.12.06	• Added *Application Interface* sub-section to *Features* Section. • Updated *EMAC Block Diagram and System Integration* section with new diagram and information. • Added *Signal Descriptions* section. • Added *EMAC Internal Interfaces* section. • Added *TX FIFO* and *RX FIFO* subsection to the *Transmit and Receive Data FIFO Buffers* section. • Updated *Descriptor Overview* section to clarify support for only enhanced (alternate) descriptors. • Added *Destination and Source Address Filtering Summary* in *Frame Filtering* Section. • Added *Clock Structure* sub-section to *Clocks and Resets* section • Added *Application Interface to FPGA Fabric* section in *Functional Description of EMAC* • Added *System Level EMAC Configuration Registers* section in *Ethernet Programming Model* • Added *EMAC Interface Initialization for FPGA GMII/MII Mode* section in Ethernet Programming Model • Added *EMAC Interface Initialization for RGMII/RMII Mode* section in Ethernet Programming Model • Corrected *DMA Initialization* and *EMAC Initialization and Configuration* titles to appear on correct initialization information • Removed duplicate programming information for DMA • Added *Taking the Ethernet MAC Out of Reset* section.
August 2014	2014.08.18	Initial release.

USB 2.0 OTG Controller 18

2016.10.28

a10_5v4 Subscribe Send Feedback

The hard processor system (HPS) provides two instances of a USB On-The-Go (OTG) controller that supports both device and host functions. The controller supports all high-speed, full-speed, and low-speed transfers in both device and host modes. The controller is fully compliant with the *On The Go and Embedded Host Supplement to the USB Revision 2.0 Specification*. The controller can be programmed for both device and host functions to support data movement over the USB protocol.

The controllers are operationally independent of each other. Each USB OTG controller supports a single USB port connected through a USB 2.0 Transceiver Macrocell Interface Plus (UTMI+) Low Pin Interface (ULPI) compliant PHY. The USB OTG controllers are instances of the Synopsys® (†)DesignWare® Cores USB 2.0 Hi-Speed On-The-Go (DWC_otg) controller.

The USB OTG controller is optimized for the following applications and systems: †

- Portable electronic devices †
- Point-to-point applications (no hub, direct connection to HS, FS, or LS device) †
- Multi-point applications (as an embedded USB host) to devices (hub and split support) †

Each of the two USB OTG ports supports both host and device modes, as described in the *On The Go and Embedded Host Supplement to the USB Revision 2.0 Specification*. The USB OTG ports support connections for all types of USB peripherals, including the following peripherals:

- Mouse
- Keyboard
- Digital cameras
- Network adapters
- Hard drives
- Generic hubs

(†) Portions © 2016 Synopsys, Inc. Used with permission. All rights reserved. Synopsys & DesignWare are registered trademarks of Synopsys, Inc. All documentation is provided "as is" and without any warranty. Synopsys expressly disclaims any and all warranties, express, implied, or otherwise, including the implied warranties of merchantability, fitness for a particular purpose, and non infringement, and any warranties arising out of a course of dealing or usage of trade.

† Paragraphs marked with the dagger (†) symbol are Synopsys Proprietary. Used with permission.

Features of the USB OTG Controller

The USB OTG controller has the following USB-specific features:

- Complies with both Revision 1.3 and Revision 2.0 of the *On The Go and Embedded Host Supplement to the USB Revision 2.0 Specification*
- Supports software-configurable modes of operation between OTG 1.3 and OTG 2.0
- Can operate in Host or Device mode
- Supports multi-point applications with hub and split support
- Supports all USB 2.0 speeds:
  - High speed (HS, 480-Mbps)
  - Full speed (FS, 12-Mbps)
  - Low speed (LS, 1.5-Mbps)

  Note: In host mode, all speeds are supported. However, in device mode, only high speed and full speed are supported.
- Integrated scatter-gather DMA supports moving data between memory and the controller
- Supports USB 2.0 in ULPI mode
- Supports all USB transaction types:
  - Control transfers
  - Bulk transfers
  - Isochronous transfers
  - Interrupts
- Supports automatic ping capability
- Supports Session Request Protocol (SRP) and Host Negotiation Protocol (HNP)
- Supports suspend, resume, and remote wake
- Supports up to 16 host channels

  Note: In host mode, when the number of device endpoints is greater than the number of host channels, software can reprogram the channels to support up to 127 devices, each having 32 endpoints (IN + OUT), for a maximum of 4,064 endpoints.
- Supports up to 16 bidirectional endpoints, including control endpoint 0

  Note: Only seven periodic device IN endpoints are supported.
- Supports a generic root hub
- Performs transaction scheduling in hardware

On the USB PHY layer, the USB OTG controller supports the following features:

- 8-bit ULPI PHY data width
- A single USB port connected to each OTG instance
- A ULPI connection to an off-chip USB transceiver
- Software-controlled access, supporting vendor-specific or optional PHY registers access to ease debug
- The OTG 2.0 support for Attach Detection Protocol (ADP) only through an external (off-chip) ADP controller

On the integration side, the USB OTG controller supports the following features:

- Different clocks for system and PHY interfaces
- Dedicated TX FIFO buffer for each device IN endpoint in direct memory access (DMA) mode
- Packet-based, dynamic FIFO memory allocation for endpoints for small FIFO buffers and flexible, efficient use of RAM that can be dynamically sized by software
- Ability to change an endpoint's FIFO memory size during transfers
- Clock gating support during USB suspend and session-off modes
  - PHY clock gating support
  - System clock gating support
- Data FIFO RAM clock gating support
- Local buffering with error correction code (ECC) support

Note: The USB OTG controller does not support the following protocols:

- Enhanced Host Controller Interface (EHCI)
- Open Host Controller Interface (OHCI)
- Universal Host Controller Interface (UHCI)

Supported PHYS

The USB OTG controller only supports USB 2.0 ULPI PHYs. Only the single data rate (SDR) mode is supported.

Refer to the *Arria 10 Device Datasheet* for specific timing information.

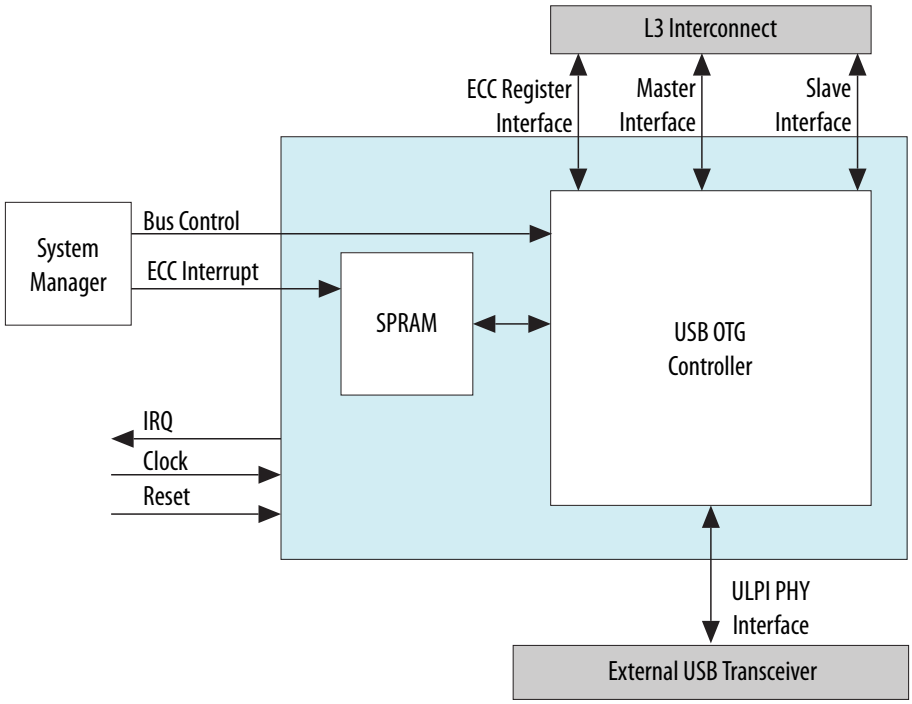
Related Information

Arria 10 Device Datasheet

USB OTG Controller Block Diagram and System Integration

Figure 18-1: USB OTG Controller System Integration

Two subsystems are included in the HPS.



The USB OTG controller connects to the level 3 (L3) interconnect through a slave interface, allowing other masters to access the control and status registers (CSRs) in the controller. The controller also connects to the L3 interconnect through a master interface, allowing the DMA engine in the controller to move data between external memory and the controller.

A single-port RAM (SPRAM) connected to the USB OTG controller is used to store USB data packets for both host and device modes. It is configured as FIFO buffers for receive and transmit data packets on the USB link.

Through the system manager, the USB OTG controller has control to use and test error correction codes (ECCs) in the SPRAM. Through the system manager, the USB OTG controller can also control the behavior of the master interface to the L3 interconnect.

The USB OTG controller connects to the external USB transceiver through a ULPI PHY interface. This interface also connects through pin multiplexers within the HPS. The pin multiplexers are controlled by the system manager.

Additional connections on the USB OTG controller include:

- Clock input from the clock manager to the USB OTG controller
- Reset input from the reset manager to the USB OTG controller
- Interrupt line from the USB OTG controller to the microprocessor unit (MPU) global interrupt controller (GIC).

The USB Controller signals are routed to the dedicated HPS pins.

Related Information

- **System Manager** on page 5-1
  Details available in the System Manager chapter.
- **General-Purpose I/O Interface** on page 22-1

USB 2.0 ULPI PHY Signal Description

Table 18-1: ULPI PHY Interfaces

The ULPI PHY interface is synchronous to the `ulpi_clk` signal coming from the PHY.

Note: For more information on how the USB signals are routed to the HPS I/O, refer to the *HPS Component Interfaces* chapter.

Port Name	Bit Width	Direction	Description
`ulpi_clk`	1	Input	ULPI Clock Receives the 60-MHz clock supplied by the high-speed ULPI PHY. All signals are synchronous to the positive edge of the clock.
`ulpi_dir`	1	Input	ULPI Data Bus Control 1—The PHY has data to transfer to the USB OTG controller. 0—The PHY does not have data to transfer.
`ulpi_nxt`	1	Input	ULPI Next Data Control Indicates that the PHY has accepted the current byte from the USB OTG controller. When the PHY is transmitting, this signal indicates that a new byte is available for the controller.
`ulpi_stp`	1	Output	ULPI Stop Data Control The controller drives this signal high to indicate the end of its data stream. The controller can also drive this signal high to request data from the PHY.

Port Name	Bit Width	Direction	Description
ulpi_data[7:0]	8	Bidirectional	Bidirectional data bus. Driven low by the controller during idle.

Related Information

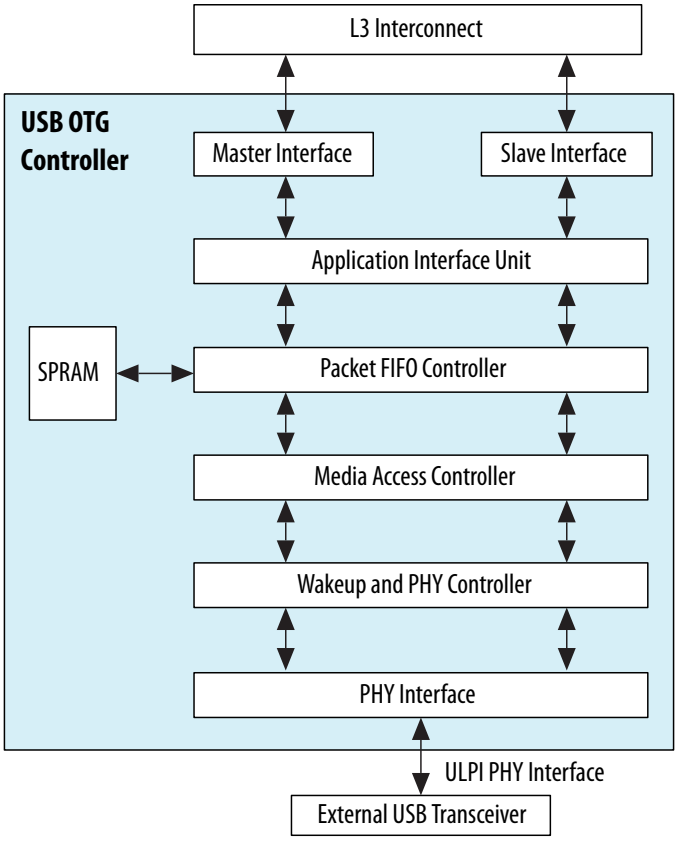
USB 2.0 OTG Controller Interface on page 28-11
For more information about how the interface signals are routed to the HPS I/O, refer to this chapter.

Functional Description of the USB OTG Controller

USB OTG Controller Block Description

Figure 18-2: USB OTG Controller Block Description

Details about each of the units that comprise the USB OTG controller are shown below.



Master Interface

The master interface includes a built-in DMA controller. The DMA controller moves data between external memory and the media access controller (MAC).

Properties of the master interface are controlled through the USB L3 Master HPROT Register (`l3master`) in the system manager. These bits provide access information to the L3 interconnect, including whether or not transactions are cacheable, bufferable, or privileged.

Note: Bits in the `l3master` register can be updated only when the master interface is guaranteed to be in an inactive state.

Slave Interface

The slave interface allows other masters in the system to access the USB OTG controller's CSRs. For testing purposes, other masters can also access the single port RAM (SPRAM).

Slave Interface CSR Unit

The slave interface can read from and write to all the CSRs in the USB OTG controllers. All register accesses are 32 bits.

The CSR is divided into the following groups of registers:

- Global
- Host
- Device
- Power and clock gating

Some registers are shared between host and device modes, because the controller can only be in one mode at a time. The controller generates a mode mismatch interrupt if a master attempts to access device registers when the controller is in host mode, or attempts to access host registers when the controller is in device mode. Writing to unimplemented registers is ignored. Reading from unimplemented registers returns indeterminate values.

Application Interface Unit

The application interface unit (AIU) generates DMA requests based on programmable FIFO buffer thresholds. The AIU generates interrupts to the GIC for both host and device modes. A DMA scheduler is included in the AIU to arbitrate and control the data transfer between packets in system memory and their respective USB endpoints.

Packet FIFO Controller

The Packet FIFO Controller (PFC) connects the AIU with the MAC through data FIFO buffers located in the SPRAM. In device mode, one FIFO buffer is implemented for each IN endpoint. In host mode, a single FIFO buffer stores data for all periodic (isochronous and interrupt) OUT endpoints, and a single FIFO buffer is used for nonperiodic (control and bulk) OUT endpoints. Host and device mode share a single receive data FIFO buffer.

SPRAM

An SPRAM implements the data FIFO buffers for host and device modes. The size of the FIFO buffers can be programmed dynamically.

The SPRAM supports ECCs.

MAC

The MAC module implements the following functionality:

- USB transaction support
- Host protocol support
- Device protocol support
- OTG protocol support

USB Transactions

In device mode, the MAC decodes and checks the integrity of all token packets. For valid OUT or SETUP tokens, the following DATA packet is also checked. If the data packet is valid, the MAC performs the following steps:

1. Writes the data to the receive FIFO buffer
2. Sends the appropriate handshake when required to the USB host

If a receive FIFO buffer is not available, the MAC sends a NAK response to the host. The MAC also supports ping protocol.

For IN tokens, if data is available in the transmit FIFO buffer, the MAC performs the following steps:

1. Reads the data from the FIFO buffer
2. Forms the data packet
3. Transmits the packet to the host
4. Receives the response from the host
5. Sends the updated status to the PFC

In host mode, the MAC receives a token request from the AIU. The MAC performs the following steps:

1. Builds the token packet
2. Sends the packet to the device

For OUT or SETUP transactions, the MAC also performs the following steps:

1. Reads the data from the transmit FIFO buffer
2. Assembles the data packet
3. Sends the packet to the device
4. Waits for a response

The response from the device causes the MAC to send a status update to the AIU.

For IN or PING transactions, the MAC waits for the data or handshake response from the device. For data responses, the MAC performs the following steps:

1. Validates the data
2. Writes the data to the receive FIFO buffer
3. Sends a status update to the AIU
4. Sends a handshake to the device, if appropriate

Host Protocol

In host mode, the MAC performs the following functions:

- Detects connect, disconnect, and remote wakeup events on the USB link
- Initiates reset
- Initiates speed enumeration processes
- Generates Start of Frame (SOF) packets.

Device Protocol

In device mode, the MAC performs the following functions:

- Handles USB reset sequence
- Handles speed enumeration
- Detects USB suspend and resume activity on the USB link
- Initiates remote wakeup
- Decodes SOF packets

OTG Protocol

The MAC handles HNP and SRP for OTG operation. HNP provides a mechanism for swapping host and device roles. SRP provides mechanisms for the host to turn off V_{BUS} to save power, and for a device to request a new USB session.

Wakeup and Power Control

To reduce power, the USB OTG controller supports a power-down mode. In power-down mode, the controller and the PHY can shut down their clocks. The controller supports wakeup on the detection of the following events:

- Resume
- Remote wakeup
- Session request protocol
- New session start

PHY Interface Unit

The USB OTG controller supports synchronous SDR data transmission to a ULPI PHY. The SDR mode implements an eight-bit data bus.

DMA

The DMA has two modes of operation. You program your software to select between scatter-gather DMA mode or buffer DMA mode depending on the controllers function.

If the controller is functioning as a generic root hub, you should program your software to select the buffer DMA mode that supports split transfers.

If generic root hub functionlaity is not required, or if the controller is configured in Device mode, you can program your software to select scatter-gather DMA mode.

If you wish to dynamically switch the mode of operation based on the queried device status or capability, your software driver must cleanly switch between the two modes of operation. For example, you may want the controller to default to scatter-gather DMA mode and only change mode when it detects a generic HUB with fast-speed and low-speed capability. Some basic requirements for switching include:

- A soft reset must be issued before changing modes.
- If buffer DMA mode is selected, then the Host mode periodic request queue depth must not be set to 16.
- Devices must be re-enumerated.

Local Memory Buffer

The NAND flash controller has three local SRAM memory buffers.

- The write FIFO buffer is a 128 × 32-bit memory (512 total bytes)
- The read FIFO buffer is a 32 × 32-bit memory (128 total bytes)
- The ECC buffer is a 96 × 16-bit memory (1536 total bytes)

The SPRAM is a 8192 x 35-bit (32 data bits and 3 control bits) memory and includes support for ECC (Error Checking and Correction). The ECC block is integrated around a memory wrapper. It provides outputs to notify the system manager when single-bit correctable errors are detected (and corrected) and when double-bit uncorrectable errors are detected. The ECC logic also allows the injection of single- and double-bit errors for test purposes. The ECC feature is disabled by default. It must be initialized to enable the ECC function.

Clocks

Table 18-2: USB OTG Controller Clock Inputs

All clocks must be operational when reset is released. No special handling is required on the clocks.

Clock Signal	Frequency	Functional Usage
usb_mp_clk	60 – 200 MHz	Drives the master and slave interfaces, DMA controller, and internal FIFO buffers
usb0_ulpi_clk	60 MHz	ULPI reference clock for usb0 from external ULPI PHY I/O pin
usb1_ulpi_clk	60 MHz	ULPI reference clock for usb1 from external ULPI PHY I/O pin

Clock Gating

You can clock gate the ulpi_clk through software. By programming the usbclken bit of the en register in the perpllgrp you can enable or disable the ulpi_clk to the USB.

Related Information

Clock Manager Address Map and Register Definitions on page 2-18

Resets

The USB OTG controller can be reset either through the hardware reset input or through software.

Reset Requirements

There must be a minimum of 12 cycles on the ulpi_clk clock before the controller is taken out of reset. During reset, the USB OTG controller asserts the ulpi_stp signal. The PHY outputs a clock when it sees

the ulpi_stp signal asserted. However, if the pin multiplexers are not programmed, the PHY does not see the ulpi_stp signal. As a result, the ulpi_clk clock signal does not arrive at the USB OTG controller.

Software must ensure that the reset is active for a minimum of two usb_mp_c lk cycles. There is no maximum assertion time.

Hardware Reset

Each of the USB OTG controllers has one reset input from the reset manager. The reset signal is asserted during a cold or warm reset event. The reset manager holds the controllers in reset until software releases the resets. Software releases resets by clearing the appropriate USB bits in the Peripheral Module Reset Register (permodrst) in the HPS reset manager.

The reset input resets the following blocks:

- The master and slave interface logic
- The integrated DMA controller
- The internal FIFO buffers
- The CSR

The reset input is synchronized to the usb_mp_clk domain. The reset input is also synchronized to the ULPI clock within the USB OTG controller and is used to reset the ULPI PHY domain logic.

Software Reset

Software can reset the controller by setting the Core Soft Reset (csftrst) bit in the Reset Register (grstctl) in the Global Registers (globgrp) group of the USB OTG controller.

Software resets are useful in the following situations:

- A PHY selection bit is changed by software. Resetting the USB OTG controller is part of clean-up to ensure that the PHY can operate with the new configuration or clock.
- During software development and debugging.

Taking the USB 2.0 OTG Controller Out of Reset

When a cold or warm reset is issued in the HPS, the reset manager resets this module and holds it in reset until software releases it.

After the Cortex-A9 MPCore CPU boots, it can deassert the reset signal by clearing the appropriate bits in the reset manager's corresponding reset register. For details about reset registers, refer to "Module Reset Signals".

Related Information

Modules Requiring Software Deassert on page 3-13

Interrupts

Table 18-3: USB OTG Interrupt Conditions

Each USB OTG controller has a single interrupt output. Interrupts are asserted on the conditions shown in the following table.

Condition	Mode
Device-initiated remote wakeup is detected.	Host mode
Session request is detected from the device.	Host mode
Device disconnect is detected.	Host mode
Host periodic TX FIFO buffer is empty (can be further programmed to indicate half-empty).	Host mode
Host channels interrupt received.	Host mode
Incomplete periodic transfer is pending at the end of the microframe.	Host mode
Host port status interrupt received.	Host mode
External host initiated resume is detected.	Device mode
Reset is detected when in suspend or normal mode.	Device mode
USB suspend mode is detected.	Device mode
Data fetch is suspended due to TX FIFO buffer full or request queue full.	Device mode
At least one isochronous OUT endpoint is pending at the end of the microframe.	Device mode
At least one isochronous IN endpoint is pending at the end of the microframe.	Device mode
At least one IN or OUT endpoint interrupt is pending at the end of the microframe.	Device mode
The end of the periodic frame is reached.	Device mode
Failure to write an isochronous OUT packet to the RX FIFO buffer. The RX FIFO buffer does not have enough space to accommodate the maximum packet size for the isochronous OUT endpoint.	Device mode
Enumeration has completed.	Device mode
Connector ID change.	Common modes
Mode mismatch. Software accesses registers belonging to an incorrect mode.	Common modes
Nonperiodic TX FIFO buffer is empty.	Common modes

Condition	Mode
RX FIFO buffer is not empty.	Common modes
Start of microframe.	Common modes
Device connection debounce is complete in host mode.	OTG interrupts
A-Device timeout while waiting for B-Device connection.	OTG interrupts
Host negotiation is complete.	OTG interrupts
Session request is complete.	OTG interrupts
Session end is detected in device mode.	OTG interrupts

USB OTG Controller Programming Model

For detailed information about using the USB OTG controller, consult your operating system (OS) driver documentation. The OS vendor provides application programming interfaces (APIs) to control USB host, device and OTG operation. This section provides a brief overview of the following software operations:

- Enabling SPRAM ECCs
- Host operation
- Device operation

Enabling SPRAM ECCs

The L3 interconnect has access to the SPRAM and is accessible through the USB OTG L3 slave interface. Software accesses the SPRAM through the `directfifo` memory space, in the USB OTG controller address space.

To enable the ECC feature, refer to the ECC chapter in the *Arria 10 Hard Processor System Technical Reference Manual*.

Note: Software cannot access the SPRAM beyond the 32-KB range. Out-of-range read transactions return indeterminate data. Out-of-range write transactions are ignored.

Related Information

Error Checking and Correction Controller on page 11-1
Error Checking and Correction (ECC) controllers provide single- and double-bit error memory protection for integrated on-chip RAM and peripheral RAMs within the hard processor system (HPS).

Host Operation

Host Initialization

After power up, the USB port is in its default mode. No VBUS is applied to the USB cable. The following process sets up the USB OTG controller as a USB host.

1. To enable power to the USB port, the software driver sets the Port Power (`prtpwr`) bit to 1 in the Host Port Control and Status Register (`hprt`) of the Host Mode Registers (`hostgrp`) group. This action drives the V_{BUS} signal on the USB link.

   The controller waits for a connection to be detected on the USB link.
2. When a USB device connects, an interrupt is generated. The Port Connect Detected (`PrtConnDet`) bit in `hprt` is set to 1.
3. Upon detecting a port connection, the software driver initiates a port reset by setting the Port Reset (`prtrst`) bit to 1 in `hprt`.
4. The software driver must wait a minimum of 10 ms so that speed enumeration can complete on the USB link.
5. After the 10 ms, the software driver sets `prtrst` back to 0 to release the port reset.
6. The USB OTG controller generates an interrupt. The Port Enable Disable Change (`prtenchng`) and Port Speed (`prtspd`) bits, in `hprt`, are set to reflect the enumerated speed of the device that attached.

   At this point the port is enabled for communication. Keep alive or SOF packets are sent on the port. If a USB 2.0-capable device fails to initialize correctly, it is reported as a USB 1.1 device.

   The Host Frame Interval Register (`hfir`) is updated with the corresponding PHY clock settings. The `hfir`, used for sending SOF packets, is in the Host Mode Registers (`host grp`) group.
7. The software driver must program the following registers in the Global Registers (`globgrp`) group, in the order listed:

   a. Receive FIFO Size Register (`grxfsiz`)—selects the size of the receive FIFO buffer
   b. Non-periodic Transmit FIFO Size Register (`gnptxfsiz`)—selects the size and the start address of the non-periodic transmit FIFO buffer for nonperiodic transactions
   c. Host Periodic Transmit FIFO Size Register (`hptxfsiz`)—selects the size and start address of the periodic transmit FIFO buffer for periodic transactions
8. System software initializes and enables at least one channel to communicate with the USB device.

Host Transaction

When configured as a host, the USB OTG controller pipes the USB transactions through one of two request queues (one for periodic transactions and one for nonperiodic transactions). Each entry in the request queue holds the SETUP, IN, or OUT channel number along with other information required to perform a transaction on the USB link. The sequence in which the requests are written to the queue determines the sequence of transactions on the USB link.

The host processes the requests in the following order at the beginning of each frame or microframe:

1. Periodic request queue, including isochronous and interrupt transactions
2. Nonperiodic request queue (bulk or control transfers)

The host schedules transactions for each enabled channel in round-robin fashion. When the host controller completes the transfer for a channel, the controller updates the DMA descriptor status in the system memory.

For OUT transactions, the host controller uses two transmit FIFO buffers to hold the packet payload to be transmitted. One transmit FIFO buffer is used for all nonperiodic OUT transactions and the other is used for all periodic OUT transactions.

For IN transactions, the USB host controller uses one receive FIFO buffer for all periodic and nonperiodic transactions. The controller holds the packet payload from the USB device in the receive FIFO buffer until the packet is transferred to the system memory. The receive FIFO buffer also holds the status of each

packet received. The status entry holds the IN channel number along with other information, including received byte count and validity status.

For generic hub operations, the USB OTG controller uses SPLIT transfers to communicate with slower-speed devices downstream of the hub. For these transfers, the transaction accumulation or buffering is performed in the generic hub, and is scheduled accordingly. The USB OTG controller ensures that enough transmit and receive buffers are allocated when the downstream transactions are completed or when accumulated data is ready to be sent upstream.

Device Operation

Device Initialization

The following process sets up the USB OTG controller as a USB device:

1. After power up, the USB OTG controller must be set to the desired device speed by writing to the Device Speed (`devspd`) bits in the Device Configuration Register (`dcfg`) in the Device Mode Registers (`devgrp`) group. After the device speed is set, the controller waits for a USB host to detect the USB port as a device port.
2. When an external host detects the USB port, the host performs a port reset, which generates an interrupt to the USB device software. The USB Reset (`usbrst`) bit in the Interrupt (`port reset`) register in the Global Registers (`globgrp`) group is set. The device software then sets up the data FIFO buffer to receive a SETUP packet from the external host. Endpoint 0 is not enabled yet.
3. After completion of the port reset, the operation speed required by the external host is known. Software reads the device speed status and sets up all the remaining required transaction fields to enable control endpoint 0.

After completion of this process, the device is receiving SOF packets, and is ready for the USB host to set up the device's control endpoint.

Device Transaction

When configured as a device, the USB OTG controller uses a single FIFO buffer to receive the data for all the OUT endpoints. The receive FIFO buffer holds the status of the received data packet, including the byte count, the data packet ID (PID), and the validity of the received data. The DMA controller reads the data out of the FIFO buffer as the data are received. If a FIFO buffer overflow condition occurs, the controller responds to the OUT packet with a NAK, and internally rewinds the pointers.

For IN endpoints, the controller uses dedicated transmit buffers for each endpoint. The application does not need to predict the order in which the USB host will access the nonperiodic endpoints. If a FIFO buffer underrun condition occurs during transmit, the controller inverts the cyclic redundancy code (CRC) to mark the packet as corrupt on the USB link.

The application handles one data packet at a time per endpoint in transaction-level operations. The software receives an interrupt on completion of every packet. Based on the handshake response received on the USB link, the application determines whether to retry the transaction or proceed with the next transaction, until all packets in the transfer are completed.

IN Transactions

For an IN transaction, the application performs the following steps:

1. Enables the endpoint
2. Triggers the DMA engine to write the associated data packet to the corresponding transmit FIFO buffer
3. Waits for the packet completion interrupt from the controller

When an IN token is received on an endpoint when the associated transmit FIFO buffer does not contain sufficient data, the controller performs the following steps:

1. Generates an interrupt
2. Returns a NAK handshake to the USB host

If sufficient data is available, the controller transmits the data to the USB host.

OUT Transactions

For an OUT transaction, the application performs the following steps:

1. Enables the endpoint
2. Waits for the packet received interrupt from the USB OTG controller
3. Retrieves the packet from the receive FIFO buffer

When an OUT token or PING token is received on an endpoint where the receive FIFO buffer does not have sufficient space, the controller performs the following steps:

1. Generates an interrupt
2. Returns a NAK handshake to USB host

If sufficient space is available, the controller stores the data in the receive FIFO buffer and returns an ACK handshake to the USB link.

Control Transfers

For control transfers, the application performs the following steps:

1. Waits for the packet received interrupt from the controller
2. Retrieves the packet from the receive buffer

Because the control transfer is governed by USB protocol, the controller always responds with an ACK handshake.

USB 2.0 OTG Controller Address Map and Register Definitions

For complete HPS address map and register definitions, refer to the **Arria 10 HPS Address Map and Register Definitions**.

Related Information

ECC Controller Address Map and Register Descriptions on page 11-18

usb_globgrp Address Map

Module Instance	Base Address	End Address
i_usbotg_0_globgrp	0xFFB00000	0xFFB003FF
i_usbotg_1_globgrp	0xFFB40000	0xFFB403FF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
gotgctl on page 18-35	0x0	32	RW	0x10000	OTG Control and Status Register
gotgint on page 18-44	0x4	32	RW	0x0	OTG Interrupt Register
gahbcfg on page 18-48	0x8	32	RW	0x0	AHB Configuration Register
gusbcfg on page 18-56	0xC	32	RW	0x1410	USB Configuration Register
grstctl on page 18-67	0x10	32	RW	0x80000000	Reset Register
gintsts on page 18-73	0x14	32	RW	0x14000020	Interrupt Register
gintmsk on page 18-87	0x18	32	RW	0x0	Interrupt Mask Register
grxstsr on page 18-93	0x1C	32	RO	0x0	Receive Status Debug Read Register
grxstsp on page 18-96	0x20	32	RO	0x0	Receive Status Read / Pop Register
grxfsiz on page 18-99	0x24	32	RW	0x2000	Receive FIFO Size Register
gnptxfsiz on page 18-100	0x28	32	RW	0x20002000	Non-periodic Transmit FIFO Size Register

Register	Offset	Width	Access	Reset Value	Description
gnptxsts on page 18-101	0x2C	32	RO	0x82000	Non-periodic Transmit FIFO/Queue Status Register
gpvndctl on page 18-105	0x34	32	RW	0x0	PHY Vendor Control Register
ggpio on page 18-108	0x38	32	RW	0x0	General Purpose Input/ Output Register
guid on page 18-108	0x3C	32	RW	0x12345678	User ID Register
gsnpsid on page 18-109	0x40	32	RO	0x4F54320A	Synopsys ID Register
ghwcfg1 on page 18-110	0x44	32	RO	0x0	User HW Config1 Register
ghwcfg2 on page 18-111	0x48	32	RO	0x208FFC90	User HW Config2 Register
ghwcfg3 on page 18-117	0x4C	32	RO	0x1F8002E8	User HW Config3 Register
ghwcfg4 on page 18-122	0x50	32	RO	0xFE0F0020	User HW Config4 Register
gdfifocfg on page 18-127	0x5C	32	RW	0x1F802000	Global DFIFO Configuration Register
hptxfsiz on page 18-128	0x100	32	RW	0x20004000	Host Periodic Transmit FIFO Size Register
dieptxf1 on page 18-129	0x104	32	RW	0x20004000	Device IN Endpoint Transmit FIFO Size Register 1
dieptxf2 on page 18-130	0x108	32	RW	0x20006000	Device IN Endpoint Transmit FIFO Size Register 2
dieptxf3 on page 18-132	0x10C	32	RW	0x20008000	Device IN Endpoint Transmit FIFO Size Register 3
dieptxf4 on page 18-133	0x110	32	RW	0x2000A000	Device IN Endpoint Transmit FIFO Size Register 4

Register	Offset	Width	Access	Reset Value	Description
dieptxf5 on page 18-134	0x114	32	RW	0x2000C000	Device IN Endpoint Transmit FIFO Size Register 5
dieptxf6 on page 18-136	0x118	32	RW	0x2000E000	Device IN Endpoint Transmit FIFO Size Register 6
dieptxf7 on page 18-137	0x11C	32	RW	0x20000000	Device IN Endpoint Transmit FIFO Size Register 7
dieptxf8 on page 18-138	0x120	32	RW	0x20002000	Device IN Endpoint Transmit FIFO Size Register 8
dieptxf9 on page 18-140	0x124	32	RW	0x20004000	Device IN Endpoint Transmit FIFO Size Register 9
dieptxf10 on page 18-141	0x128	32	RW	0x20006000	Device IN Endpoint Transmit FIFO Size Register 10
dieptxf11 on page 18-142	0x12C	32	RW	0x20008000	Device IN Endpoint Transmit FIFO Size Register 11
dieptxf12 on page 18-144	0x130	32	RW	0x2000A000	Device IN Endpoint Transmit FIFO Size Register 12
dieptxf13 on page 18-145	0x134	32	RW	0x2000C000	Device IN Endpoint Transmit FIFO Size Register 13
dieptxf14 on page 18-146	0x138	32	RW	0x2000E000	Device IN Endpoint Transmit FIFO Size Register 14
dieptxf15 on page 18-148	0x13C	32	RW	0x20000000	Device IN Endpoint Transmit FIFO Size Register 15

usb_globgrp Summary

Module Instance	Base Address
i_usbotg_0_globgrp	0xFFB00000
i_usbotg_1_globgrp	0xFFB40000

Register Address Offset	Bit Fields															
i_usbotg_0_globgrp																
gotgctl 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved										curmod RO 0x0	otgver RW 0x0	bsesvld RO 0x0	asesvld RO 0x0	dbnctime RO 0x0	conidsts RO 0x1
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ehen RW 0x0	Reserved		devhnpen RW 0x0	hstsethnpen RW 0x0	hnpreq RW 0x0	hstnegscs RO 0x0	bvalidoval RW 0x0	bvalidoven RW 0x0	avalidoval RW 0x0	avalidoven RW 0x0	vbvalidoval RW 0x0	vbvalidoven RW 0x0	sesreq RW 0x0	sesreqscs RO 0x0
gotgint 0x4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved												dbncedone RW 0x0	adevtoutchg RW 0x0	hstnegdet RW 0x0	Reserved
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						hstnegsucstschng RW 0x0	sesreqsucstschng RW 0x0	Reserved					sesenddet RW 0x0	Reserved	
gahbcfg 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved							invdescendianess RW 0x0	ahbsingle RW 0x0	notialldmawrit RW 0x0	remmemsupp RW 0x0	Reserved				
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							ptxfemplvl RW 0x0	nptxfemplvl RW 0x0	Reserved	dmaen RW 0x0	hbstlen RW 0x0				glblintrmsk RW 0x0

Register Address Offset	Bit Fields															
gusbcfg 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	corruptt xpkt WO 0x0	forcedev mode RW 0x0	forcehst mode RW 0x0	txenddel ay RW 0x0	Reserved	ic_usbcap RO 0x0	ulpi RW 0x0	indicator RW 0x0	complement RW 0x0	termseldlpulse RW 0x0	ulpiextvbusindicator RW 0x0	ulpiextvbusdrv RW 0x0	ulpiclksusm RW 0x0	ulpiautores RW 0x0	Reserved	
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		usbtrdtim RW 0x5				hnpcap RW 0x0	srpcap RW 0x0	ddrsel RW 0x0	physel RO 0x0	fsintf RO 0x0	ulpi_utmi_sel RO 0x1	phyif RO 0x0	toutcal RW 0x0		
grstctl 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ahbidle RO 0x1	dmareq RO 0x0	Reserved													
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved					txfnum RW 0x0					txfflsh RW 0x0	rxfflsh RW 0x0	Reserved	frmcntrrst RW 0x0	piufssftrst RW 0x0	csftrst RW 0x0
gintsts 0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	wkupint RW 0x0	sessreqint RW 0x0	disconnint RW 0x0	conidstschng RW 0x1	Reserved	ptxfemp RO 0x1	hchint RO 0x0	prtint RO 0x0	resetdet RW 0x0	fetsusp RW 0x0	incomplp RW 0x0	incompisoin RW 0x0	oepint RO 0x0	iepint RO 0x0	epmis RW 0x0	Reserved
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	eopf RW 0x0	isooutdrop RW 0x0	enumdone RW 0x0	usbrst RW 0x0	usbsusp RW 0x0	erlysusp RW 0x0	Reserved		goutnakeff RO 0x0	ginnakeff RO 0x0	nptxfemp RO 0x1	rxflvl RO 0x0	sof RW 0x0	otgint RO 0x0	modemis RW 0x0	curmod RO 0x0

Register Address Offset	Bit Fields															
gintmsk 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	wkupintmsk RW 0x0	sessreqintmsk RW 0x0	disconnintmsk RW 0x0	connidstschngmsk RW 0x0	Reserved	ptxfempmsk RW 0x0	hchintmsk RW 0x0	prtintmsk RW 0x0	resetdetmsk RW 0x0	fetsuspmsk RW 0x0	incomplpmsk RW 0x0	incompisoinmsk RW 0x0	oepintmsk RW 0x0	iepintmsk RW 0x0	epmismsk RW 0x0	Reserved
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	eopfmsk RW 0x0	isooutdropmsk RW 0x0	enumdonemsk RW 0x0	usbrstmsk RW 0x0	usbsuspmsk RW 0x0	erlysuspmsk RW 0x0	Reserved		goutnakeffmsk RW 0x0	ginnakeffmsk RW 0x0	Reserved	rxflvlmsk RW 0x0	sofmsk RW 0x0	otgintmsk RW 0x0	modemismsk RW 0x0	Reserved
grxstsr 0x1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved							fn RO 0x0				pktsts RO 0x0				dpid RO 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	dpid RO 0x0	bcnt RO 0x0											chnum RO 0x0			
grxstsp 0x20	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved							fn RO 0x0				pktsts RO 0x0				dpid RO 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	dpid RO 0x0	bcnt RO 0x0											chnum RO 0x0			
grxfsiz 0x24	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	rxfdep RW 0x2000														
gnptxfsiz 0x28	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	nptxfdep RW 0x2000															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	nptxfstaddr RW 0x2000															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gnptxsts 0x2C	Reserved	nptxqtop RO 0x0							nptxqspcavail RO 0x8							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	nptxfspcavail RO 0x2000															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gpvndctl 0x34	disulpidrvr RW 0x0	Reserved			vstsdone RW 0x0	vstsbsy RO 0x0	newregreq RW 0x0	Reserved		regwr RW 0x0	regaddr RW 0x0					
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	vctrl RW 0x0								regdata RW 0x0							
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ggpio 0x38	gpo RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	gpi RO 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
guid 0x3C	guid RO 0x12345678															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	guid RO 0x12345678															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gsnpsid 0x40	gsnpsid RO 0x4F54320A															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	gsnpsid RO 0x4F54320A															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ghwcfg1	ghwcfg1 RO 0x0															
0x44	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ghwcfg1 RO 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ghwcfg2	Reserved	tknqdepth RO 0x8					ptxqdepth RO 0x0		nptxqdepth RO 0x2		Reserved	multiprocintrpt RO 0x0	dynfifosizing RO 0x1	periosupport RO 0x1	numhstchnl RO 0xF	
0x48	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	numhstchnl RO 0xF	numdeveps RO 0xF					fsphytype RO 0x0		hsphytype RO 0x2		singpnt RO 0x0	otgarch RO 0x2		otgmode RO 0x0		
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ghwcfg3	dfifodepth RO 0x1F80															
0x4C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	lpmmode RO 0x0	bcsupport RO 0x0	hsicmode RO 0x0	adpsupport RO 0x0	rsttype RO 0x0	optfeature RO 0x0	vndctlsupt RO 0x1	i2cintsel RO 0x0	otgen RO 0x1	pktsizewidth RO 0x6			xfersizewidth RO 0x8			
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ghwcfg4	dma RO 0x1	dma_configuration RO 0x1	ineps RO 0xF				dedfifomode RO 0x1	sessendfltr RO 0x0	bvalidfltr RO 0x0	avalidfltr RO 0x0	vbusvalidfltr RO 0x0	iddgfltr RO 0x0	numctleps RO 0xF			
0x50	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	phydata-width RO 0x0	Reserved							extendedhibernation RO 0x0	hibernation RO 0x0	ahbfreq RO 0x1	partialpwrdn RO 0x0	numdevperioeps RO 0x0			

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gdfifocfg 0x5C	epinfobaseaddr RW 0x1F80															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	gdfifocfg RW 0x2000															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hptxfsiz 0x100	Reserved		ptxfsize RW 0x2000													
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		ptxfstaddr RW 0x4000													
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dieptxf1 0x104	Reserved		inepntxfdep RW 0x2000													
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		inepntxfstaddr RW 0x4000													
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dieptxf2 0x108	Reserved		inepntxfdep RW 0x2000													
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		inepntxfstaddr RW 0x6000													
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dieptxf3 0x10C	Reserved		inepntxfdep RW 0x2000													
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	inepntxfstaddr RW 0x8000															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dieptxf4 0x110	Reserved		inepntxfdep RW 0x2000													
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	inepntxfstaddr RW 0xA000															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dieptxf5	Reserved		inepntxfdep RW 0x2000													
0x114	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	inepntxfstaddr RW 0xC000															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dieptxf6	Reserved		inepntxfdep RW 0x2000													
0x118	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	inepntxfstaddr RW 0xE000															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dieptxf7	Reserved		inepntxfdep RW 0x2000													
0x11C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	inepntxfstaddr RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dieptxf8	Reserved		inepntxfdep RW 0x2000													
0x120	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	inepntxfstaddr RW 0x2000															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dieptxf9	Reserved		inepntxfdep RW 0x2000													
0x124	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	inepntxfstaddr RW 0x4000															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dieptxf10	Reserved		inepntxfdep RW 0x2000													
0x128	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	inepntxfstaddr RW 0x6000															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dieptxf11 0x12C	Reserved		inepntxfdep RW 0x2000													
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	inepntxfstaddr RW 0x8000															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dieptxf12 0x130	Reserved		inepntxfdep RW 0x2000													
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	inepntxfstaddr RW 0xA000															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dieptxf13 0x134	Reserved		inepntxfdep RW 0x2000													
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	inepntxfstaddr RW 0xC000															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dieptxf14 0x138	Reserved		inepntxfdep RW 0x2000													
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	inepntxfstaddr RW 0xE000															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dieptxf15 0x13C	Reserved		inepntxfdep RW 0x2000													
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	inepntxfstaddr RW 0x0															
i_usbotg_1_globgrp																

Register Address Offset	Bit Fields															
gotgctl 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved										curmod RO 0x0	otgver RW 0x0	bsesvld RO 0x0	asesvld RO 0x0	dbnctime RO 0x0	conidsts RO 0x1
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	ehen RW 0x0	Reserved		devhnpen RW 0x0	hstsethnpen RW 0x0	hnpreq RW 0x0	hstnegscs RO 0x0	bvalidoval RW 0x0	bvalidoven RW 0x0	avalidoval RW 0x0	avalidoven RW 0x0	vbvalidoval RW 0x0	vbvalidoven RW 0x0	sesreq RW 0x0	sesreqscs RO 0x0
gotgint 0x4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved												dbncedone RW 0x0	adevtoutchg RW 0x0	hstnegdet RW 0x0	Reserved
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						hstnegsucstschng RW 0x0	sesreqsucstschng RW 0x0	Reserved					sesenddet RW 0x0	Reserved	
gahbcfg 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved							invdescendianess RW 0x0	ahbsingle RW 0x0	notialldmawrit RW 0x0	remmemsupp RW 0x0	Reserved				
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							ptxfemplvl RW 0x0	nptxfemplvl RW 0x0	Reserved	dmaen RW 0x0	hbstlen RW 0x0				glblintrmsk RW 0x0

Register Address Offset	Bit Fields															
gusbcfg 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	corruptt xpkt WO 0x0	forcedev mode RW 0x0	forcehst mode RW 0x0	txenddel ay RW 0x0	Reserved	ic_usbcap RO 0x0	ulpi RW 0x0	indicator RW 0x0	complement RW 0x0	termseldlpulse RW 0x0	ulpiextvbusindicator RW 0x0	ulpiextvbusdrv RW 0x0	ulpiclksusm RW 0x0	ulpiautores RW 0x0	Reserved	
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		usbtrdtim RW 0x5				hnpcap RW 0x0	srpcap RW 0x0	ddrsel RW 0x0	physel RO 0x0	fsintf RO 0x0	ulpi_utmi_sel RO 0x1	phyif RO 0x0	toutcal RW 0x0		
grstctl 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ahbidle RO 0x1	dmareq RO 0x0	Reserved													
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved					txfnum RW 0x0					txfflsh RW 0x0	rxfflsh RW 0x0	Reserved	frmcntrrst RW 0x0	piufssftrst RW 0x0	csftrst RW 0x0
gintsts 0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	wkupint RW 0x0	sessreqint RW 0x0	disconnint RW 0x0	conidstschng RW 0x1	Reserved	ptxfemp RO 0x1	hchint RO 0x0	prtint RO 0x0	resetdet RW 0x0	fetsusp RW 0x0	incomplp RW 0x0	incompisoin RW 0x0	oepint RO 0x0	iepint RO 0x0	epmis RW 0x0	Reserved
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	eopf RW 0x0	isooutdrop RW 0x0	enumdone RW 0x0	usbrst RW 0x0	usbsusp RW 0x0	erlysusp RW 0x0	Reserved		goutnakeff RO 0x0	ginnakeff RO 0x0	nptxfemp RO 0x1	rxflvl RO 0x0	sof RW 0x0	otgint RO 0x0	modemis RW 0x0	curmod RO 0x0

Register Address Offset	Bit Fields															
gintmsk 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	wkupintmsk RW 0x0	sessreqintmsk RW 0x0	disconnintmsk RW 0x0	conidstschngmsk RW 0x0	Reserved	ptxfempmsk RW 0x0	hchintmsk RW 0x0	prtintmsk RW 0x0	resetdetmsk RW 0x0	fetsuspmsk RW 0x0	incomplpmsk RW 0x0	incompisoinmsk RW 0x0	oepintmsk RW 0x0	iepintmsk RW 0x0	epmismsk RW 0x0	Reserved
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	eopfmsk RW 0x0	isooutdropmsk RW 0x0	enumdonemsk RW 0x0	usbrstmsk RW 0x0	usbsuspmsk RW 0x0	erlysuspmsk RW 0x0	Reserved		goutnakeffmsk RW 0x0	ginnakeffmsk RW 0x0	Reserved	rxflvlmsk RW 0x0	sofmsk RW 0x0	otgintmsk RW 0x0	modemismsk RW 0x0	Reserved
grxstsr 0x1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved							fn RO 0x0				pktsts RO 0x0				dpid RO 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	dpid RO 0x0	bcnt RO 0x0											chnum RO 0x0			
grxstsp 0x20	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved							fn RO 0x0				pktsts RO 0x0				dpid RO 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	dpid RO 0x0	bcnt RO 0x0											chnum RO 0x0			
grxfsiz 0x24	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	rxfdep RW 0x2000														
gnptxfsiz 0x28	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	nptxfdep RW 0x2000															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	nptxfstaddr RW 0x2000															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gnptxsts	Reserved	nptxqtop RO 0x0							nptxqspcavail RO 0x8							
0x2C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	nptxfspcavail RO 0x2000															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gpvndctl	dislpidrvr RW 0x0	Reserved			vstsdone RW 0x0	vstsbsy RO 0x0	newregreq RW 0x0	Reserved		regwr RW 0x0	regaddr RW 0x0					
0x34	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	vctrl RW 0x0								regdata RW 0x0							
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ggpio	gpo RW 0x0															
0x38	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	gpi RO 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
guid	guid RO 0x12345678															
0x3C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	guid RO 0x12345678															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gsnpsid	gsnpsid RO 0x4F54320A															
0x40	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	gsnpsid RO 0x4F54320A															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ghwcfg1 0x44	ghwcfg1 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ghwcfg1 RO 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ghwcfg2 0x48	Reserved	tknqdepth RO 0x8					ptxqdepth RO 0x0		nptxqdepth RO 0x2		Reserved	multiprocintrpt RO 0x0	dynfifosizing RO 0x1	periosupport RO 0x1	numhstchnl RO 0xF	
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	numhstchnl RO 0xF		numdeveps RO 0xF				fsphytype RO 0x0		hsphytype RO 0x2		singpnt RO 0x0	otgarch RO 0x2		otgmode RO 0x0		
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ghwcfg3 0x4C	dfifodepth RO 0x1F80															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	lpmmode RO 0x0	bcsupport RO 0x0	hsicmode RO 0x0	adpsupport RO 0x0	rsttype RO 0x0	optfeature RO 0x0	vndctlsupt RO 0x1	i2cintsel RO 0x0	otgen RO 0x1	pktsizewidth RO 0x6			xfersizewidth RO 0x8			
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ghwcfg4 0x50	dma RO 0x1	dma_configuration RO 0x1	ineps RO 0xF				dedfifomode RO 0x1	sessendfltr RO 0x0	bvalidfltr RO 0x0	avalidfltr RO 0x0	vbusvalidfltr RO 0x0	iddgfltr RO 0x0	numctleps RO 0xF			
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	phydata-width RO 0x0	Reserved							extendedhibernation RO 0x0	hibernation RO 0x0	ahbfreq RO 0x1	partialpwrdn RO 0x0	numdevperioeps RO 0x0			

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gdfifocfg 0x5C	epinfobaseaddr RW 0x1F80															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	gdfifocfg RW 0x2000															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hptxfsiz 0x100	Reserved		ptxfsize RW 0x2000													
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		ptxfstaddr RW 0x4000													
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dieptxf1 0x104	Reserved		inepntxfdep RW 0x2000													
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		inepntxfstaddr RW 0x4000													
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dieptxf2 0x108	Reserved		inepntxfdep RW 0x2000													
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		inepntxfstaddr RW 0x6000													
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dieptxf3 0x10C	Reserved		inepntxfdep RW 0x2000													
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	inepntxfstaddr RW 0x8000															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dieptxf4 0x110	Reserved		inepntxfdep RW 0x2000													
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	inepntxfstaddr RW 0xA000															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dieptxf5	Reserved		inepntxfdep RW 0x2000													
0x114	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	inepntxfstaddr RW 0xC000															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dieptxf6	Reserved		inepntxfdep RW 0x2000													
0x118	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	inepntxfstaddr RW 0xE000															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dieptxf7	Reserved		inepntxfdep RW 0x2000													
0x11C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	inepntxfstaddr RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dieptxf8	Reserved		inepntxfdep RW 0x2000													
0x120	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	inepntxfstaddr RW 0x2000															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dieptxf9	Reserved		inepntxfdep RW 0x2000													
0x124	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	inepntxfstaddr RW 0x4000															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dieptxf10	Reserved		inepntxfdep RW 0x2000													
0x128	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	inepntxfstaddr RW 0x6000															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved		inepntxfdep RW 0x2000													
dieptxf11 0x12C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	inepntxfstaddr RW 0x8000															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved		inepntxfdep RW 0x2000													
dieptxf12 0x130	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	inepntxfstaddr RW 0xA000															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved		inepntxfdep RW 0x2000													
dieptxf13 0x134	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	inepntxfstaddr RW 0xC000															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved		inepntxfdep RW 0x2000													
dieptxf14 0x138	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	inepntxfstaddr RW 0xE000															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved		inepntxfdep RW 0x2000													
dieptxf15 0x13C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	inepntxfstaddr RW 0x0															

gotgctl

OTG Control and Status Register

Module Instance	Base Address	Register Address
i_usbotg_0_globgrp	0xFFB00000	0xFFB00000
i_usbotg_1_globgrp	0xFFB40000	0xFFB40000

Offset: 0x0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved										curmod RO 0x0	otgver RW 0x0	bsesvld RO 0x0	asesvld RO 0x0	dbnctime RO 0x0	conidsts RO 0x1
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	ehen RW 0x0	Reserved		devhnpen RW 0x0	hstsethnpen RW 0x0	hnpreq RW 0x0	hstnegscs RO 0x0	bvalidovval RW 0x0	bvalidoven RW 0x0	avalidovval RW 0x0	avalidoven RW 0x0	vbvalidovval RW 0x0	vbvalidoven RW 0x0	sesreq RW 0x0	sesreqscs RO 0x0

gotgctl Fields

Bit	Name	Description	Access	Reset
21	curmod	Mode: Host and Device Current Mode of Operation (CurMod) Indicates the current mode. 1'b0: Device mode 1'b1: Host mode	RO	0x0

Bit	Name	Description	Access	Reset
20	otgver	OTG Version (OTGVer) Indicates the OTG revision. 1'b0: OTG Version 1.3. In this version the core supports Data line pulsing and VBus pulsing for SRP. 1'b1: OTG Version 2.0. In this version the core supports only Data line pulsing for SRP. **Value** **Description** 0x0 VER13 0x1 VER20	RW	0x0
19	bsesvld	Mode: Device only B-Session Valid (BSesVld) Indicates the Device mode transceiver status. 1'b0: B-session is not valid. 1'b1: B-session is valid. In OTG mode, you can use this bit to determine IF the device is connected or disconnected. Note: If you do not enabled OTG features (such as SRP and HNP), the read reset value will be 1.The vbus assigns the values internally for non- SRP or non-HNP configurations. **Value** **Description** 0x0 NOTVALID 0x1 VALID	RO	0x0

Bit	Name	Description	Access	Reset
18	asesvld	Mode: Host only A-Session Valid (ASesVld) Indicates the Host mode transceiver status. 1'b0: A-session is not valid 1'b1: A-session is valid Note: If you do not enabled OTG features (such as SRP and HNP), the read reset value will be 1.The vbus assigns the values internally for non- SRP or non-HNP configurations. **Value** **Description** 0x0 VALID 0x1 NOTVALID	RO	0x0
17	dbnctime	Mode: Host only Long/Short Debounce Time (DbncTime) Indicates the debounce time of a detected connection. 1'b0: Long debounce time, used FOR physical connections (100 ms + 2.5 micro-sec) 1'b1: Short debounce time, used FOR soft connections (2.5 micro-sec) **Value** **Description** 0x0 LONG 0x1 SHORT	RO	0x0
16	conidsts	Mode: Host and Device Connector ID Status (ConIDSts) Indicates the connector ID status on a connect event. 1'b0: The DWC_otg core is in A-Device mode 1'b1: The DWC_otg core is in B-Device mode **Value** **Description** 0x0 MODEA 0x1 MODEB	RO	0x1

Bit	Name	Description	Access	Reset
14	ehen	Mode: SRP Capable Host Embedded Host Enable (EHEn) 1'b1 : Enable Embedded Host Mode. 1'b0: Disable Embedded Host Mode.	RW	0x0
11	devhnpen	Mode: Device only Device HNP Enabled (DevHNPEn) The application sets this bit when it successfully receives a SetFeature.SetHNPEnable command from the connected USB host. 1'b0: HNP is not enabled in the application 1'b1: HNP is enabled in the application **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
10	hstsethnpen	Mode: Host only Host Set HNP Enable (HstSetHNPEn) The application sets this bit when it has successfully enabled HNP (using the SetFeature.SetHNPEnable command) on the connected device. 1'b0: Host Set HNP is not enabled 1'b1: Host Set HNP is enabled **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
9	hnpreq	Mode: Device only HNP Request (HNPReq) The application sets this bit to initiate an HNP request to the connected USB host. The application can clear this bit by writing a 0 when the Host Negotiation Success Status Change bit in the OTG Interrupt register (GOTGINT.HstNegSucStsChng) is SET. The core clears this bit when the HstNegSucStsChng bit is cleared. 1'b0: No HNP request 1'b1: HNP request **Value: Description** 0x0: DISABLED 0x1: ENABLED	RW	0x0
8	hstnegscs	Mode: Device only Host Negotiation Success (HstNegScs) The core sets this bit when host negotiation is successful. The core clears this bit when the HNP Request (HNPReq) bit in this register is SET. 1'b0: Host negotiation failure 1'b1: Host negotiation success **Value: Description** 0x0: FAIL 0x1: SUCCESS	RO	0x0

Bit	Name	Description	Access	Reset
7	bvalidovval	B-Peripheral Session Valid Override-Value (BvalidOvVal) This bit is used to set Override value for Bvalid signal when GOTGCTL.BvalidOvEn is set. 1'b0 : Bvalid value is 1'b0 when GOTGCTL.BvalidOvEn =1 1'b1 : Bvalid value is 1'b1 when GOTGCTL.BvalidOvEn =1 **Value** **Description** 0x0 VALUE0 0x1 VALUE1	RW	0x0
6	bvalidoven	B-Peripheral Session Valid Override Enable (BvalidOvEn) This bit is used to enable/disable the software to override the Bvalid signal using the GOTGCTL.BvalidOvVal. 1'b1 : Internally Bvalid received from the PHY is overridden with GOTGCTL.BvalidOvVal. 1'b0 : Override is disabled and bvalid signal from the respective PHY selected is used internally by the force **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
5	avalidovval	A-Peripheral Session Valid Override-Value (AvalidOvVal) This bit is used to set Override value for Avalid signal when GOTGCTL.AvalidOvEn is set. 1'b0 : Avalid value is 1'b0 when GOTGCTL.AvalidOvEn =1 1'b1 : Avalid value is 1'b1 when GOTGCTL.AvalidOvEn =1 **Value** **Description** 0x0 VALUE0 0x1 VALUE1	RW	0x0
4	avalidoven	A-Peripheral Session Valid Override Enable (AvalidOvEn) This bit is used to enable/disable the software to override the Avalid signal using the GOTGCTL.AvalidOvVal. 1'b1 : Internally Avalid received from the PHY is overridden with GOTGCTL.AvalidOvVal. 1'b0 : Override is disabled and avalid signal from the respective PHY selected is used internally by the core **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
3	vbvalidovval	VBUS Valid OverrideValue (VbvalidOvVal) This bit is used to set Override value for vbusvalid signal when GOTGCTL.VbvalidOvEn is set. 1'b0 : vbusvalid value is 1'b0 when GOTGCTL.VbvalidOvEn =1 1'b1 : vbusvalid value is 1'b1 when GOTGCTL.VbvalidOvEn =1 **Value** **Description** 0x0 SET0 0x1 SET1	RW	0x0
2	vbvalidoven	VBUS Valid Override Enable (VbvalidOvEn) This bit is used to enable/disable the software to override the Bvalid signal using the GOTGCTL.VbvalidOvVal. 1'b1 : Internally Bvalid received from the PHY is overridden with GOTGCTL.VbvalidOvVal. 1'b0 : Override is disabled and bvalid signal from the respective PHY selected is used internally by the core **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
1	sesreq	Mode: Device only Session Request (SesReq) The application sets this bit to initiate a session request on the USB. The application can clear this bit by writing a 0 when the Host Negotiation Success Status Change bit in the OTG Interrupt register (GOTGINT.HstNegSucStsChng) is SET. The core clears this bit when the HstNegSucStsChng bit is cleared. If you use the USB 1.1 Full-Speed Serial Transceiver interface to initiate the session request, the application must wait until the VBUS discharges to 0.2 V, after the B-Session Valid bit in this register (GOTGCTL.BSesVld) is cleared. This discharge time varies between different PHYs and can be obtained from the PHY vendor. 1'b0: No session request 1'b1: Session request **Value: Description** 0x0: NOREQUEST 0x1: REQUEST	RW	0x0
0	sesreqscs	Mode: Device only Session Request Success (SesReqScs) The core sets this bit when a session request initiation is successful. 1'b0: Session request failure 1'b1: Session request success **Value: Description** 0x0: FAIL 0x1: SUCCESS	RO	0x0

gotgint

OTG Interrupt Register

Module Instance	Base Address	Register Address
i_usbotg_0_globgrp	0xFFB00000	0xFFB00004
i_usbotg_1_globgrp	0xFFB40000	0xFFB40004

Offset: 0x4

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved												dbncedone RW 0x0	adevtoutchg RW 0x0	hstnegdet RW 0x0	Reserved
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						hstnegsuccstschng RW 0x0	sesreqsuccstschng RW 0x0	Reserved					sesenddet RW 0x0	Reserved	

gotgint Fields

Bit	Name	Description	Access	Reset
19	dbncedone	Mode: Host only Debounce Done (DbnceDone) The core sets this bit when the debounce is completed after the device connect. The application can start driving USB reset after seeing this interrupt. This bit is only valid when the HNP Capable or SRP Capable bit is SET in the Core USB Configuration register (GUSBCFG.HNPCap or GUSBCFG.SRPCap, respectively).This bit can be set only by the core and the application should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
18	adevtoutchg	Mode:Host and Device A-Device Timeout Change (ADevTOUTChg) The core sets this bit to indicate that the A-device has timed out WHILE waiting FOR the B-device to connect.This bit can be set only by the core and the application should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
17	hstnegdet	Mode:Host and Device Host Negotiation Detected (HstNegDet) The core sets this bit when it detects a host negotiation request on the USB.This bit can be set only by the core and the application should write 1 to clear it. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
9	hstnegsucstschng	Mode:Host and Device Host Negotiation Success Status Change (HstNegSucStsChng) The core sets this bit on the success or failure of a USB host negotiation request. The application must read the Host Negotiation Success bit of the OTG Control and Status register (GOTGCTL.HstNegScs) to check For success or failure.This bit can be set only by the core and the application should write 1 to clear it. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
8	sesreqsucstschng	Mode:Host and Device Session Request Success Status Change (SesReqSucStsChng) The core sets this bit on the success or failure of a session request. The application must read the Session Request Success bit in the OTG Control and Status register (GOTGCTL.SesReqScs) to check For success or failure.This bit can be set only by the core and the application should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
2	sesenddet	Mode:Host and Device Session End Detected (SesEndDet) The core sets this bit when the utmiotg_bvalid signal is deasserted.This bit can be set only by the core and the application should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

gahbcfg

AHB Configuration Register

Module Instance	Base Address	Register Address
i_usbotg_0_globgrp	0xFFB00000	0xFFB00008
i_usbotg_1_globgrp	0xFFB40000	0xFFB40008

Offset: 0x8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							invdescendianess RW 0x0	ahbsingle RW 0x0	notialldmawrit RW 0x0	remmemsupp RW 0x0	Reserved				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							ptxfemplvl RW 0x0	nptxfemplvl RW 0x0	Reserved	dmaen RW 0x0	hbstlen RW 0x0				glblintrmsk RW 0x0

gahbcfg Fields

Bit	Name	Description	Access	Reset
24	invdescendianess	Invert Descriptor Endianess (InvDescEndianess) 1'b0: Descriptor Endianness is same as AHB Master Endianness 1'b1: Descriptor Endianness is Little Endian if AHB Master Endianness is Big Endian. Descriptor Endianness is Big Endian if AHB Master Endianness is Little Endian.	RW	0x0

Bit	Name	Description	Access	Reset
23	ahbsingle	AHB Single Support (AHBSingle) This bit when programmed supports Single transfers for the remaining data in a transfer when the DWC_otg core is operating in DMA mode. 1'b0: This is the default mode. When this bit is set to 1'b0, the remaining data in the transfer is sent using INCR burst size. 1'b1: When set to 1'b1, the remaining data in a transfer is sent using Single burst size. Note: if this feature is enabled, the AHB RETRY and SPLIT transfers still have INCR burst type. Enable this feature when the AHB Slave connected to the DWC_otg core does not support INCR burst (and when Split, and Retry transactions are not being used in the bus).	RW	0x0

Bit	Name	Description	Access	Reset
22	notialldmawrit	Notify All Dma Write Transactions (NotiAllDmaWrit) This bit is programmed to enable the System DMA Done functionality for all the DMA write Transactions corresponding to the Channel/ Endpoint. This bit is valid only when GAHBCFG.RemMemSupp is set to 1. GAHBCFG.NotiAllDmaWrit = 1 - HSOTG core asserts int_dma_req for all the DMA write transactions on the AHB interface along with int_ dma_done, chep_last_transact and chep_number signal informations. The core waits for sys_dma_done signal for all the DMA write transactions in order to complete the transfer of a particular Channel/Endpoint. GAHBCFG.NotiAllDmaWrit = 0 - HSOTG core asserts int_dma_req signal only for the last transac- tion of DMA write transfer corresponding to a particular Channel/Endpoint. Similarly, the core waits for sys_ dma_done signal only for that transaction of DMA write to complete the transfer of a particular Channel/Endpoint. **Value** **Description** 0x0 LASTTRANS 0x1 ALLTRANS	RW	0x0

Bit	Name	Description	Access	Reset
21	remmemsupp	Remote Memory Support (RemMemSupp) This bit is programmed to enable the functionality to wait for the system DMA Done Signal for the DMA Write Transfers. GAHBCFG.RemMemSupp=1 - The int_dma_req output signal is asserted when HSOTG DMA starts write transfer to the external memory. When the core is done with the Transfers it asserts int_dma_done signal to flag the completion of DMA writes from HSOTG. The core then waits for sys_dma_done signal from the system to proceed further and complete the Data Transfer corresponding to a particular Channel/Endpoint. GAHBCFG.RemMemSupp=0 - The int_dma_req and int_dma_done signals are not asserted and the core proceeds with the assertion of the XferComp interrupt as soon as the DMA write transfer is done at the HSOTG Core Boundary and it doesn't wait for the sys_dma_done signal to complete the DATA transfers **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
8	ptxfemplvl	Mode:Host only Periodic TxFIFO Empty Level (PTxFEmpLvl) Indicates when the Periodic TxFIFO Empty Interrupt bit in the Core Interrupt register (GINTSTS.PTxFEmp) is triggered. This bit is used only in Slave mode. 1'b0: GINTSTS.PTxFEmp interrupt indicates that the Periodic TxFIFO is half empty 1'b1: GINTSTS.PTxFEmp interrupt indicates that the Periodic TxFIFO is completely empty **Value** **Description** 0x0 HALFEMPTY 0x1 EMPTY	RW	0x0

Bit	Name	Description	Access	Reset
7	nptxfemplvl	Mode:Host and device Non-Periodic TxFIFO Empty Level (NPTxFEmpLvl) This bit is used only in Slave mode. In host mode and with Shared FIFO with device mode, this bit indicates when the Non-Periodic TxFIFO Empty Interrupt bit in the Core Interrupt register (GINTSTS.NPTxFEmp) is triggered. With dedicated FIFO in device mode, this bit indicates when IN endpoint Transmit FIFO empty interrupt (DIEPINTn.TxFEmp) is triggered. Host mode and with Shared FIFO with device mode:- 1'b0: GINTSTS.NPTxFEmp interrupt indicates that the Non-Periodic TxFIFO is half empty 1'b1: GINTSTS.NPTxFEmp interrupt indicates that the Non-Periodic TxFIFO is completely empty Dedicated FIFO in device mode :- 1'b0: DIEPINTn.TxFEmp interrupt indicates that the IN Endpoint TxFIFO is half empty 1'b1: DIEPINTn.TxFEmp interrupt indicates that the IN Endpoint TxFIFO is completely empty **Value** **Description** 0x0 HALFEMPTY 0x1 EMPTY	RW	0x0
5	dmaen	Mode:Host and device DMA Enable (DMAEn) 1'b0: Core operates in Slave mode 1'b1: Core operates in a DMA mode This bit is always 0 when Slave-Only mode has been selected **Value** **Description** 0x0 SLAVEMODE 0x1 DMAMODE	RW	0x0

Bit	Name	Description	Access	Reset
4:1	hbstlen	Mode:Host and device Burst Length/Type (HBstLen) This field is used in both External and Internal DMA modes. In External DMA mode, these bits appear on dma_burst[3:0] ports, which can be used by an external wrapper to interface the External DMA Controller interface to Synopsys DW_ahb_dmac or ARM PrimeCell. External DMA Modedefines the DMA burst length in terms of 32-bit words: 4'b0000: 1 word 4'b0001: 4 words 4'b0010: 8 words 4'b0011: 16 words 4'b0100: 32 words 4'b0101: 64 words 4'b0110: 128 words 4'b0111: 256 words Others: Reserved Internal DMA ModeAHB Master burst type: 4'b0000 Single 4'b0001 INCR 4'b0011 INCR4 4'b0101 INCR8 4'b0111 INCR16 Others: Reserved Value — Description 0x0 — WORD1ORSINGLE 0x1 — WORD4ORINCR 0x2 — WORD8 0x3 — WORD16ORINCR4 0x4 — WORD32 0x5 — WORD64ORINCR8 0x6 — WORD128 0x7 — WORD256ORINCR16 0x8 — WORDX	RW	0x0

Bit	Name	Description	Access	Reset
0	glblintrmsk	Mode:Host and device Global Interrupt Mask (GlblIntrMsk) The application uses this bit to mask or unmask the interrupt line assertion to itself. Irrespective of this bit's setting, the interrupt status registers are updated by the core. 1'b0: Mask the interrupt assertion to the application. 1'b1: Unmask the interrupt assertion to the application. **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0

gusbcfg

USB Configuration Register

Module Instance	Base Address	Register Address
i_usbotg_0_globgrp	0xFFB00000	0xFFB0000C
i_usbotg_1_globgrp	0xFFB40000	0xFFB4000C

Offset: 0xC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
corruptt xpkt WO 0x0	force devmo de RW 0x0	force hstmo de RW 0x0	txend delay RW 0x0	Reser ved	ic_ usbca p RO 0x0	ulpi RW 0x0	indic ator RW 0x0	compl ement RW 0x0	terms eldlp ulse RW 0x0	ulpie xtvbu sindi cator RW 0x0	ulpie xtvbu sdrv RW 0x0	ulpic lksus m RW 0x0	ulpia utore s RW 0x0	Reserved	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		usbtrdtim RW 0x5				hnpca p RW 0x0	srpca p RW 0x0	ddrse l RW 0x0	physe l RO 0x0	fsint f RO 0x0	ulpi_ utmi_ sel RO 0x1	phyif RO 0x0	toutcal RW 0x0		

gusbcfg Fields

Bit	Name	Description	Access	Reset
31	corrupttxpkt	Mode:Host and device Corrupt Tx packet This bit is FOr debug purposes only. Never Set this bit to 1.The application should always write 1'b0 to this bit. **Value** — **Description** 0x0 — NODEBUG 0x1 — DEBUG	WO	0x0
30	forcedevmode	Mode:Host and device Force Device Mode (ForceDevMode) Writing a 1 to this bit forces the core to device mode irrespective of utmiotg_iddig input pin. 1'b0 : Normal Mode. 1'b1 : Force Device Mode. After setting the force bit, the application must wait at least 25 ms before the change to take effect. When the simulation is in scale down mode, waiting for 500 micro sec is sufficient. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
29	forcehstmode	Mode:Host and device Force Host Mode (ForceHstMode) Writing a 1 to this bit forces the core to host mode irrespective of utmiotg_iddig input pin. 1'b0 : Normal Mode. 1'b1 : Force Host Mode. After setting the force bit, the application must wait at least 25 ms before the change to take effect. When the simulation is in scale down mode, waiting for 500 micro sec is sufficient. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
28	txenddelay	Mode: Device only Tx End Delay (TxEndDelay) Writing 1'b1 to this bit enables the core to follow the TxEndDelay timings as per UTMI+ specification 1.05 section 4.1.5 for opmode signal during remote wakeup. 1'b0 : Normal Mode. 1'b1 : Tx End delay. **Value** **Description** 0x0 DISABLED	RW	0x0
26	ic_usbcap	IC_USB-Capable (IC_USBCap) The application uses this bit to control the DWC_otg core's IC_USB capabilities. 1'b0: IC_USB PHY Interface is not selected. 1'b1: IC_USB PHY Interface is selected. This bit is writable only if IC_USB is selected	RO	0x0

Bit	Name	Description	Access	Reset
25	ulpi	Mode:Host only ULPI Interface Protect Disable Controls circuitry built into the PHY For protecting the ULPI interface when the link tri-states STP and data. Any pull-ups or pull-downs employed by this feature can be disabled. Please refer to the ULPI Specification For more detail. 1'b0: Enables the interface protect circuit 1'b1: Disables the interface protect circuit **Value** **Description** 0x0 ENABLED 0x1 DISABLED	RW	0x0
24	indicator	Mode:Host only Indicator Pass Through Controls wether the Complement Output is qualified with the Internal Vbus Valid comparator before being used in the Vbus State in the RX CMD. Please refer to the ULPI Spec for more detail. 1'b0: Complement Output signal is qualified with the Internal VbusValid comparator. 1'b1: Complement Output signal is not qualified with the Internal VbusValid comparator. **Value** **Description** 0x0 QUALIFIED 0x1 NONQUALIFIED	RW	0x0

Bit	Name	Description	Access	Reset
23	complement	Mode:Host only Indicator Complement Controls the PHY to invert the ExternalVbusIndicator input signal, generating the Complement Output. Please refer to the ULPI Spec For more detail 1'b0: PHY does not invert External-VbusIndicator signal 1'b1: PHY does invert ExternalVbu-sIndicator signal **Value: Description** 0x0: NONINVERT 0x1: INVERT	RW	0x0
22	termseldlpulse	Mode:Device only TermSel DLine Pulsing Selection (TermSelDLPulse) This bit selects utmi_termselect to drive data line pulse during SRP. 1'b0: Data line pulsing using utmi_txvalid (Default). 1'b1: Data line pulsing using utmi_termsel. **Value: Description** 0x0: TXVALID 0x1: TERMSEL	RW	0x0
21	ulpiextvbusindicator	Mode:Host only ULPI External VBUS Indicator (ULPIExtVbusIndicator) This bit indicates to the ULPI PHY to use an external VBUS overcurrent indicator. 1'b0: PHY uses internal VBUS valid comparator. 1'b1: PHY uses external VBUS valid comparator. **Value: Description** 0x0: INTERN 0x1: EXTERN	RW	0x0

Bit	Name	Description	Access	Reset
20	ulpiextvbusdrv	Mode:Host only ULPI External VBUS Drive (ULPIExtV-busDrv) This bit selects between internal or external supply to drive 5V on VBUS, in ULPI PHY. 1'b0: PHY drives VBUS using internal charge pump (Default). 1'b1: PHY drives VBUS using external supply. **Value** **Description** 0x0 INTERN 0x1 EXTERN	RW	0x0
19	ulpiclksusm	Mode:Host and Device ULPI Clock SuspendM (ULPIClkSusM) This bit sets the ClockSuspendM bit in the Interface Control register on the ULPI PHY. This bit applies only in serial or carkit modes. 1'b0: PHY powers down internal clock during suspend. 1'b1: PHY does not power down internal clock. **Value** **Description** 0x0 PWDCLK 0x1 NONPWDCLK	RW	0x0
18	ulpiautores	Mode:Host and Device ULPI Auto Resume (ULPIAutoRes) This bit sets the AutoResume bit in the Interface Control register on the ULPI PHY. 1'b0: PHY does not use AutoResume feature. 1'b1: PHY uses AutoResume feature. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
13:10	usbtrdtim	Mode: Device only USB Turnaround Time (USBTrdTim) Sets the turnaround time in PHY clocks. Specifies the response time For a MAC request to the Packet FIFO Controller (PFC) to fetch data from the DFIFO (SPRAM). This must be programmed to 4'h5: When the MAC interface is 16-bit UTMI+ . 4'h9: When the MAC interface is 8-bit UTMI+ . Note: The values above are calculated For the minimum AHB frequency of 30 MHz. USB turnaround time is critical For certification where long cables and 5-Hubs are used, so If you need the AHB to run at less than 30 MHz, and If USB turnaround time is not critical, these bits can be programmed to a larger value. **Value** **Description** 0x9 TURNTIME	RW	0x5
9	hnpcap	Mode:Host and Device HNP-Capable (HNPCap) The application uses this bit to control the DWC_otg core's HNP capabilities. 1'b0: HNP capability is not enabled. 1'b1: HNP capability is enabled. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
8	srpcap	Mode:Host and Device SRP-Capable (SRPCap) The application uses this bit to control the DWC_otg core SRP capabilities. If the core operates as a non-SRP-capable B-device, it cannot request the connected A-device (host) to activate VBUS and start a session. 1'b0: SRP capability is not enabled. 1'b1: SRP capability is enabled. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
7	ddrsel	Mode:Host and Device ULPI DDR Select (DDRSel) The application uses this bit to select a Single Data Rate (SDR) or Double Data Rate (DDR) or ULPI interface. 1'b0: Single Data Rate ULPI Interface, with 8-bit-wide data bus 1'b1: Double Data Rate ULPI Interface, with 4-bit-wide data bus **Value** **Description** 0x0 SDR 0x1 DDR	RW	0x0

Bit	Name	Description	Access	Reset
6	physel	Mode:Host and Device USB 2.0 High-Speed PHY or USB 1.1 Full-Speed Serial Transceiver Select (PHYSel) The application uses this bit to select either a high-speed UTMI+ or ULPI PHY, or a full-speed transceiver. 1'b0: USB 2.0 high-speed UTMI+ or ULPI PHY 1'b1: USB 1.1 full-speed serial transceiver If a USB 1.1 Full-Speed Serial Transceiver interface was not selected in, this bit is always 0, with Write Only access. If a high-speed PHY interface was not selected in, this bit is always 1, with Write Only access. If both interface types were selected (parameters have non-zero values), the application uses this bit to select which interface is active, and access is Read and Write. **Value** **Description** 0x0 USB20	RO	0x0

Bit	Name	Description	Access	Reset
5	fsintf	Mode:Host and Device Full-Speed Serial Interface Select (FSIntf) The application uses this bit to select either a unidirectional or bidirectional USB 1.1 full-speed serial transceiver interface. 1'b0: 6-pin unidirectional full-speed serial interface 1'b1: 3-pin bidirectional full-speed serial interface If a USB 1.1 Full-Speed Serial Transceiver interface was not selected, this bit is always 0, with Write Only access. If a USB 1.1 FS interface was selected, Then the application can Set this bit to select between the 3- and 6-pin interfaces, and access is Read and Write. **Value** **Description** 0x0 FS6PIN 0x1 FS3PIN	RO	0x0
4	ulpi_utmi_sel	Mode:Host and Device ULPI or UTMI+ Select (ULPI_UTMI_Sel) The application uses this bit to select either a UTMI+ interface or ULPI Interface. 1'b0: UTMI+ Interface 1'b1: ULPI Interface This bit is writable only If UTMI+ and ULPI was specified For High-Speed PHY Interface(s). **Value** **Description** 0x0 ULPI	RO	0x1

Bit	Name	Description	Access	Reset
3	phyif	Mode:Host and Device PHY Interface (PHYIf) The application uses this bit to configure the core To support a UTMI+ PHY with an 8- or 16-bit interface. When a ULPI PHY is chosen, this must be Set to 8-bit mode. 1'b0: 8 bits 1'b1: 16 bits This bit is writable only If UTMI+ and ULPI were selected **Value** **Description** 0x0 BITS8	RO	0x0
2:0	toutcal	Mode:Host and Device HS/FS Timeout Calibration (TOutCal) The number of PHY clocks that the application programs in this field is added to the high-speed/ full-speed interpacket timeout duration in the core to account For any additional delays introduced by the PHY. This can be required, because the delay introduced by the PHY in generating the linestate condition can vary from one PHY to another. The USB standard timeout value For high-speed operation is 736 to 816 (inclusive) bit times. The USB standard timeout value For full-speed operation is 16 to 18 (inclusive) bit times. The application must program this field based on the speed of enumeration. The number of bit times added per PHY clock are: High-speed operation: One 30-MHz PHY clock = 16 bit times One 60-MHz PHY clock = 8 bit times Full-speed operation: One 30-MHz PHY clock = 0.4 bit times One 60-MHz PHY clock = 0.2 bit times One 48-MHz PHY clock = 0.25 bit times	RW	0x0

grstctl

Reset Register

Module Instance	Base Address	Register Address
i_usbotg_0_globgrp	0xFFB00000	0xFFB00010
i_usbotg_1_globgrp	0xFFB40000	0xFFB40010

Offset: 0x10

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ahbidle RO 0x1	dmareq RO 0x0	Reserved													
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					txfnum RW 0x0					txfflsh RW 0x0	rxfflsh RW 0x0	Reserved	frmcntrrst RW 0x0	piufssftrst RW 0x0	csftrst RW 0x0

grstctl Fields

Bit	Name	Description	Access	Reset
31	ahbidle	Mode:Host and Device AHB Master Idle (AHBIdle) Indicates that the AHB Master State Machine is in the IDLE condition. Value — Description 0x0 — INACTIVE 0x1 — ACTIVE	RO	0x1

Bit	Name	Description	Access	Reset
30	dmareq	Mode:Host and Device DMA Request Signal (DMAReq) Indicates that the DMA request is in progress. Used For debug. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0

Bit	Name	Description	Access	Reset
10:6	txfnum	Mode:Host and Device TxFIFO Number (TxFNum) This is the FIFO number that must be flushed using the TxFIFO Flush bit. This field must not be changed until the core clears the TxFIFO Flush bit. 5'h0: - Non-periodic TxFIFO flush in Host mode - Non-periodic TxFIFO flush in device mode when in shared FIFO operation - Tx FIFO 0 flush in device mode when in dedicated FIFO mode 5'h1: - Periodic TxFIFO flush in Host mode - Periodic TxFIFO 1 flush in Device mode when in shared FIFO operation - TXFIFO 1 flush in device mode when in dedicated FIFO mode 5'h2: - Periodic TxFIFO 2 flush in Device mode when in shared FIFO operation - TXFIFO 2 flush in device mode when in dedicated FIFO mode ... 5'hF: - Periodic TxFIFO 15 flush in Device mode when in shared FIFO operation - TXFIFO 15 flush in device mode when in dedicated FIFO mode 5'h10: Flush all the transmit FIFOs in device or host mode. **Value** **Description** 0xf TXF15 0x0 TXF0 0x1 TXF1 0x2 TXF2 0x10 TXF16	RW	0x0

Bit	Name	Description	Access	Reset
5	txfflsh	Mode:Host and Device TxFIFO Flush (TxFFlsh) This bit selectively flushes a single or all transmit FIFOs, but cannot do so If the core is in the midst of a transaction. The application must write this bit only after checking that the core is neither writing to the TxFIFO nor reading from the TxFIFO. Verify using these registers: ReadNAK Effective Interrupt ensures the core is not reading from the FIFO WriteGRSTCTL.AHBIdle ensures the core is not writing anything to the FIFO. Flushing is normally recommended when FIFOs are reconfigured or when switching between Shared FIFO and Dedicated Transmit FIFO operation. FIFO flushing is also recommended during device endpoint disable. The application must wait until the core clears this bit before performing any operations. This bit takes eight clocks to clear, using the slower clock of phy_clk or hclk. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
4	rxfflsh	Mode:Host and Device RxFIFO Flush (RxFFlsh) The application can flush the entire RxFIFO using this bit, but must first ensure that the core is not in the middle of a transaction. The application must only write to this bit after checking that the core is neither reading from the RxFIFO nor writing to the RxFIFO. The application must wait until the bit is cleared before performing any other operations. This bit requires 8 clocks (slowest of PHY or AHB clock) to clear. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
2	frmcntrrst	Mode:Host only Host Frame Counter Reset (FrmCntrRst) The application writes this bit to reset the (micro)frame number counter inside the core. When the (micro)frame counter is reset, the subsequent SOF sent out by the core has a (micro)frame number of 0.When application writes 1 to the bit, it might not be able to read back the value as it will get cleared by the core in a few clock cycles. **Value** **Description** 0x0 NOTACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
1	piufssftrst	Mode:Host and Device PIU FS Dedicated Controller Soft Reset (PIUFSSftRst) Resets the PIU FS Dedicated Controller All module state machines in FS Dedicated Controller of PIU are reset to the IDLE state. Used to reset the FS Dedicated controller in PIU in case of any PHY Errors like Loss of activity or Babble Error resulting in the PHY remaining in RX state for more than one frame boundary	RW	0x0
0	csftrst	Mode:Host and Device Core Soft Reset (CSftRst) Resets the hclk and phy_clock domains as follows: Clears the interrupts and all the CSR registers except the following register bits: • PCGCCTL.RstPdwnModule • PCGCCTL.GateHclk • PCGCCTL.PwrClmp • PCGCCTL.StopPPhyLPwrClkSelclk • GUSBCFG.PhyLPwrClkSel • GUSBCFG.DDRSel • GUSBCFG.PHYSel • GUSBCFG.FSIntf • GUSBCFG.ULPI_UTMI_Sel • GUSBCFG.PHYIf • GUSBCFG.TxEndDelay • GUSBCFG.TermSelDLPulse • GUSBCFG.ULPIClkSusM • GUSBCFG.ULPIAutoRes • GUSBCFG.ULPIFsLs • GGPIO • GPWRDN • GADPCTL • HCFG.FSLSPclkSel • DCFG.DevSpd • DCTL.SftDiscon • All module state machines All module state machines (except the AHB Slave Unit) are reset to the IDLE state, and all the transmit FIFOs and the receive FIFO are flushed.	RW	0x0

Bit	Name	Description	Access	Reset
		Any transactions on the AHB Master are terminated as soon as possible, after gracefully completing the last data phase of an AHB transfer. Any transactions on the USB are terminated immediately. When Hibernation or ADP feature is enabled, the PMU module is not reset by the Core Soft Reset. The application can write to this bit any time it wants to reset the core. This is a self-clearing bit and the core clears this bit after all the necessary logic is reset in the core, which can take several clocks, depending on the current state of the core. Once this bit is cleared software must wait at least 3 PHY clocks before doing any access to the PHY domain (synchronization delay). Software must also must check that bit 31 of this register is 1 (AHB Master is IDLE) before starting any operation. Typically software reset is used during software development and also when you dynamically change the PHY selection bits in the USB configuration registers listed above. When you change the PHY, the corresponding clock For the PHY is selected and used in the PHY domain. Once a new clock is selected, the PHY domain has to be reset for proper operation. **Value** **Description** 0x0 NOTACTIVE 0x1 ACTIVE		

gintsts

Interrupt Register

Module Instance	Base Address	Register Address
i_usbotg_0_globgrp	0xFFB00000	0xFFB00014
i_usbotg_1_globgrp	0xFFB40000	0xFFB40014

Offset: 0x14

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
wkupint RW 0x0	sessreqint RW 0x0	disconnint RW 0x0	conidstschng RW 0x1	Reserved	ptxfemp RO 0x1	hchint RO 0x0	prtint RO 0x0	resetdet RW 0x0	fetsusp RW 0x0	incomplp RW 0x0	incompisoin RW 0x0	oepint RO 0x0	iepint RO 0x0	epmis RW 0x0	Reserved
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
eopf RW 0x0	isoutdrop RW 0x0	enumdone RW 0x0	usbrst RW 0x0	usbusp RW 0x0	erlysusp RW 0x0	Reserved		goutnakeff RO 0x0	ginnakeff RO 0x0	nptxfemp RO 0x1	rxflvl RO 0x0	sof RW 0x0	otgint RO 0x0	modemis RW 0x0	curmod RO 0x0

gintsts Fields

Bit	Name	Description	Access	Reset
31	wkupint	Mode:Host and Device Resume/Remote Wakeup Detected Interrupt (WkUpInt) Wakeup Interrupt during Suspend(L2) or LPM(L1) state. During Suspend(L2): - Device Mode - This interrupt is asserted only when Host Initiated Resume is detected on USB. - Host Mode - This interrupt is asserted only when Device Initiated Remote Wakeup is detected on USB. For more information, see 'Partial Power-Down and Clock Gating Programming Model'. During LPM(L1):- - Device Mode - This interrupt is asserted for either Host Initiated Resume or Device Initiated Remote Wakeup on USB. - Host Mode - This interrupt is asserted for either Host Initiated Resume or Device Initiated Remote Wakeup on USB. For more information, see 'LPM Entry and Exit Programming Model' **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
30	sessreqint	Mode:Host and Device Session Request/New Session Detected Interrupt (SessReqInt) In Host mode, this interrupt is asserted when a session request is detected from the device. In Host mode, this interrupt is asserted when a session request is detected from the device. In Device mode, this interrupt is asserted when the utmisrp_bvalid signal goes high. For more information on how to use this interrupt, see 'Partial Power-Down and Clock Gating Programming Model'. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
29	disconnint	Mode:Host only Disconnect Detected Interrupt (DisconnInt) Asserted when a device disconnect is detected. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
28	conidstschng	Mode:Host and Device Connector ID Status Change (ConIDStsChng) The core sets this bit when there is a change in connector ID status. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x1

Bit	Name	Description	Access	Reset
26	ptxfemp	Mode:Host only Periodic TxFIFO Empty (PTxFEmp) This interrupt is asserted when the Periodic Transmit FIFO is either half or completely empty and there is space for at least one entry to be written in the Periodic Request Queue. The half or completely empty status is determined by the Periodic TxFIFO Empty Level bit in the Core AHB Configuration register (GAHBCFG.PTxFEmpLvl). **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RO	0x1
25	hchint	Mode:Host only Host Channels Interrupt (HChInt) The core sets this bit to indicate that an interrupt is pending on one of the channels of the core (in Host mode). The application must read the Host All Channels Interrupt (HAINT) register to determine the exact number of the channel on which the interrupt occurred, and Then read the corresponding Host Channel-n Interrupt (HCINTn) register to determine the exact cause of the interrupt. The application must clear the appropriate status bit in the HCINTn register to clear this bit. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RO	0x0

Bit	Name	Description	Access	Reset
24	prtint	Mode:Host only Host Port Interrupt (PrtInt) The core sets this bit to indicate a change in port status of one of the DWC_otg core ports in Host mode. The application must read the Host Port Control and Status (HPRT) register to determine the exact event that caused this interrupt. The application must clear the appropriate status bit in the Host Port Control and Status register to clear this bit. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RO	0x0
23	resetdet	Mode: Device only Reset detected Interrupt (ResetDet) In Device mode, this interrupt is asserted when a reset is detected on the USB in partial power-down mode when the device is in Suspend. In Host mode, this interrupt is not asserted. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
22	fetsusp	Mode: Device only Data Fetch Suspended (FetSusp) This interrupt is valid only in DMA mode. This interrupt indicates that the core has stopped fetching data For IN endpoints due to the unavailability of TxFIFO space or Request Queue space. This interrupt is used by the application For an endpoint mismatch algorithm. For example, after detecting an endpoint mismatch, the application: Sets a Global non-periodic IN NAK handshake Disables In endpoints Flushes the FIFO Determines the token sequence from the IN Token Sequence Learning Queue Re-enables the endpoints Clears the Global non-periodic IN NAK handshake If the Global non-periodic IN NAK is cleared, the core has not yet fetched data For the IN endpoint, and the IN token is received: the core generates an 'IN token received when FIFO empty' interrupt. The OTG Then sends the host a NAK response. To avoid this scenario, the application can check the GINTSTS.FetSusp interrupt, which ensures that the FIFO is full before clearing a Global NAK handshake. Alternatively, the application can mask the “IN token received when FIFO empty” interrupt when clearing a Global IN NAK handshake. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
21	incomplp	Incomplete Periodic Transfer (incomplP) In Host mode, the core sets this interrupt bit when there are incomplete periodic transactions still pending which are scheduled For the current microframe. Incomplete Isochronous OUT Transfer (incompISOOUT) The Device mode, the core sets this interrupt to indicate that there is at least one isochronous OUT endpoint on which the transfer is not completed in the current microframe. This interrupt is asserted along with the End of Periodic Frame Interrupt (EOPF) bit in this register. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
20	incompisoin	Mode: Device only Incomplete Isochronous IN Transfer (incompISOIN) The core sets this interrupt to indicate that there is at least one isochronous IN endpoint on which the transfer is not completed in the current microframe. This interrupt is asserted along with the End of Periodic Frame Interrupt (EOPF) bit in this register. Note: This interrupt is not asserted in Scatter/Gather DMA mode. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
19	oepint	Mode: Device only OUT Endpoints Interrupt (OEPInt) The core sets this bit to indicate that an interrupt is pending on one of the OUT endpoints of the core (in Device mode). The application must read the Device All Endpoints Interrupt (DAINT) register to determine the exact number of the OUT endpoint on which the interrupt occurred, and Then read the corresponding Device OUT Endpoint-n Interrupt (DOEPINTn) register to determine the exact cause of the interrupt. The application must clear the appropriate status bit in the corresponding DOEPINTn register to clear this bit. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0
18	iepint	Mode: Device only IN Endpoints Interrupt (IEPInt) The core sets this bit to indicate that an interrupt is pending on one of the IN endpoints of the core (in Device mode). The application must read the Device All Endpoints Interrupt (DAINT) register to determine the exact number of the IN endpoint on Device IN Endpoint-n Interrupt (DIEPINTn) register to determine the exact cause of the interrupt. The application must clear the appropriate status bit in the corresponding DIEPINTn register to clear this bit. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0

Bit	Name	Description	Access	Reset
17	epmis	Mode: Device only Endpoint Mismatch Interrupt (EPMis) Note: This interrupt is valid only in shared FIFO operation. Indicates that an IN token has been received For a non-periodic endpoint, but the data For another endpoint is present in the top of the Non-periodic Transmit FIFO and the IN endpoint mismatch count programmed by the application has expired. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
15	eopf	Mode: Device only End of Periodic Frame Interrupt (EOPF) Indicates that the period specified in the Periodic Frame Interval field of the Device Configuration register (DCFG.PerFrInt) has been reached in the current microframe.	RW	0x0
14	isooutdrop	Mode: Device only Isochronous OUT Packet Dropped Interrupt (ISOOutDrop) The core sets this bit when it fails to write an isochronous OUT packet into the RxFIFO because the RxFIFO does not have enough space to accommodate a maximum packet size packet for the isochronous OUT endpoint. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
13	enumdone	Mode: Device only Enumeration Done (EnumDone) The core sets this bit to indicate that speed enumeration is complete. The application must read the Device Status (DSTS) register to obtain the enumerated speed. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
12	usbrst	Mode: Device only USB Reset (USBRst) The core sets this bit to indicate that a reset is detected on the USB. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
11	usbsusp	Mode: Device only USB Suspend (USBSusp) The core sets this bit to indicate that a suspend was detected on the USB. The core enters the Suspended state when there is no activity on the linestate signal For an extended period of time. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
10	erlysusp	Mode: Device only Early Suspend (ErlySusp) The core sets this bit to indicate that an Idle state has been detected on the USB For 3 ms. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
7	goutnakeff	Mode: Device only Global OUT NAK Effective (GOUTNakEff) Indicates that the Set Global OUT NAK bit in the Device Control register (DCTL.SGOUTNak), Set by the application, has taken effect in the core. This bit can be cleared by writing the Clear Global OUT NAK bit in the Device Control register (DCTL.CGOUTNak). **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RO	0x0

Bit	Name	Description	Access	Reset
6	ginnakeff	Mode: Device only Global IN Non-periodic NAK Effective (GINNakEff) Indicates that the Set Global Non-periodic IN NAK bit in the Device Control register (DCTL.SGNPInNak), Set by the application, has taken effect in the core. That is, the core has sampled the Global IN NAK bit Set by the application. This bit can be cleared by clearing the Clear Global Non-periodic IN NAK bit in the Device Control register (DCTL.CGNPInNak). This interrupt does not necessarily mean that a NAK handshake is sent out on the USB. The STALL bit takes precedence over the NAK bit. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0
5	nptxfemp	Mode: Host and Device Non-periodic TxFIFO Empty (NPTxFEmp) This interrupt is asserted when the Non-periodic TxFIFO is either half or completely empty, and there is space For at least one Entry to be written to the Non-periodic Transmit Request Queue. The half or completely empty status is determined by the Non-periodic TxFIFO Empty Level bit in the Core AHB Configuration register (GAHBCFG.NPTxFEmpLvl).	RO	0x1
4	rxflvl	Mode: Host and Device RxFIFO Non-Empty (RxFLvl) Indicates that there is at least one packet pending to be read from the RxFIFO. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0

Bit	Name	Description	Access	Reset
3	sof	Mode: Host and Device Start of (micro)Frame (Sof) In Host mode, the core sets this bit to indicate that an SOF (FS), micro-SOF (HS), or Keep-Alive (LS) is transmitted on the USB. The application must write a 1 to this bit to clear the interrupt. In Device mode, in the core sets this bit to indicate that an SOF token has been received on the USB. The application can read the Device Status register to get the current (micro)Frame number. This interrupt is seen only when the core is operating at either HS or FS. Note: This register may return 1'b1 if read immediately after power on reset. If the register bit reads 1'b1 immediately after power on reset it does not indicate that an SOF has been sent (in case of host mode) or SOF has been received (in case of device mode). The read value of this interrupt is valid only after a valid connection between host and device is established. If the bit is set after power on reset the application can clear the bit. **Value** — **Description** 0x0 — INTACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
2	otgint	Mode: Host and Device OTG Interrupt (OTGInt) The core sets this bit to indicate an OTG protocol event. The application must read the OTG Interrupt Status (GOTGINT) register to determine the exact event that caused this interrupt. The application must clear the appropriate status bit in the GOTGINT register to clear this bit. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0
1	modemis	Mode: Host and Device Mode Mismatch Interrupt (ModeMis) The core sets this bit when the application is trying to access: A Host mode register, when the core is operating in Device mode A Device mode register, when the core is operating in Host mode The register access is completed on the AHB with an OKAY response, but is ignored by the core internally and does not affect the operation of the core.This bit can be set only by the core and the application should write 1 to clear it **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
0	curmod	Mode: Host and Device Current Mode of Operation (CurMod) Indicates the current mode. 1'b0: Device mode 1'b1: Host mode Value: 0x0 — Description: DEVICE Value: 0x1 — Description: HOST	RO	0x0

gintmsk

Interrupt Mask Register

Module Instance	Base Address	Register Address
i_usbotg_0_globgrp	0xFFB00000	0xFFB00018
i_usbotg_1_globgrp	0xFFB40000	0xFFB40018

Offset: 0x18

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
wkupintmsk RW 0x0	sessreqintmsk RW 0x0	disconnintmsk RW 0x0	conidstschngmsk RW 0x0	Reserved	ptxfempmsk RW 0x0	hchintmsk RW 0x0	prtintmsk RW 0x0	resetdetmsk RW 0x0	fetsuspmsk RW 0x0	incomplpmsk RW 0x0	incompisoinmsk RW 0x0	oepintmsk RW 0x0	iepintmsk RW 0x0	epmismsk RW 0x0	Reserved
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
eopfmsk RW 0x0	isoutdropmsk RW 0x0	enumdonemsk RW 0x0	usbrstmsk RW 0x0	usbuspmsk RW 0x0	erlyuspmsk RW 0x0	Reserved		goutnakeffmsk RW 0x0	ginnakeffmsk RW 0x0	Reserved	rxflvlmsk RW 0x0	sofmsk RW 0x0	otgintmsk RW 0x0	modemismsk RW 0x0	Reserved

gintmsk Fields

Bit	Name	Description	Access	Reset
31	wkupintmsk	Mode: Host and Device Resume/Remote Wakeup Detected Interrupt Mask The WakeUp bit is used for LPM state wake up in a way similar to that of wake up in suspend state. (WkUpIntMsk) **Value** — **Description** 0x0 — MASK 0x1 — NOMASK	RW	0x0
30	sessreqintmsk	Mode: Host and Device Session Request/New Session Detected Interrupt Mask (SessReqIntMsk) **Value** — **Description** 0x0 — MASK 0x1 — NOMASK	RW	0x0
29	disconnintmsk	Mode: Host and Device Disconnect Detected Interrupt Mask (DisconnIntMsk) **Value** — **Description** 0x0 — MASK 0x1 — NOMASK	RW	0x0
28	conidstschngmsk	Mode: Host and Device Connector ID Status Change Mask (ConIDStsChngMsk) **Value** — **Description** 0x0 — MASK 0x1 — NOMASK	RW	0x0

Bit	Name	Description	Access	Reset
26	ptxfempmsk	Mode: Host only Periodic TxFIFO Empty Mask (PTxFEmpMsk) **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0
25	hchintmsk	Mode: Host only Host Channels Interrupt Mask (HChIntMsk) **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0
24	prtintmsk	Mode: Host only Host Port Interrupt Mask (PrtIntMsk) **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0
23	resetdetmsk	Mode: Device only Reset detected Interrupt Mask (ResetDetMsk) **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0
22	fetsuspmsk	Mode: Device only Data Fetch Suspended Mask (FetSuspMsk) **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0

Bit	Name	Description	Access	Reset
21	incomplpmsk	Mode: Host only Incomplete Periodic Transfer Mask (incomplPMsk) Mode: Device only Incomplete Isochronous OUT Transfer Interrupt Mask (incompISOOUTMsk) **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0
20	incompisoinmsk	Mode: Device only Incomplete Isochronous IN Transfer Mask (incompISOINMsk) **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0
19	oepintmsk	Mode: Device only OUT Endpoints Interrupt Mask (OEPIntMsk) **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0
18	iepintmsk	Mode: Device only IN Endpoints Interrupt Mask (IEPIntMsk) **Value** **Description** 0x0 MASK 0x1 NOMAKS	RW	0x0
17	epmismsk	Mode: Device only Endpoint Mismatch Interrupt Mask (EPMisMsk) **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0

Bit	Name	Description	Access	Reset
15	eopfmsk	Mode: Device only End of Periodic Frame Interrupt Mask (EOPFMsk) **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0
14	isooutdropmsk	Mode: Device only Isochronous OUT Packet Dropped Interrupt Mask (ISOOutDropMsk) **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0
13	enumdonemsk	Mode: Device only Enumeration Done Mask (EnumDoneMsk) **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0
12	usbrstmsk	Mode: Device only USB Reset Mask (USBRstMsk) **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0
11	usbsuspmsk	Mode: Device only USB Suspend Mask (USBSuspMsk) **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0

Bit	Name	Description	Access	Reset
10	erlysuspmsk	Mode: Device only Early Suspend Mask (ErlySuspMsk) **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0
7	goutnakeffmsk	Mode: Device only Global OUT NAK Effective Mask (GOUTNakEffMsk) **Value** **Description** 0x0 MASK 0x1 NOMAKS	RW	0x0
6	ginnakeffmsk	Mode: Device only Global Non-periodic IN NAK Effective Mask (GINNakEffMsk) **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0
4	rxflvlmsk	Mode: Host and Device Receive FIFO Non-Empty Mask (RxFLvlMsk) **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0
3	sofmsk	Mode: Host and Device Start of (micro)Frame Mask (SofMsk) **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0

Bit	Name	Description	Access	Reset
2	otgintmsk	Mode: Host and Device OTG Interrupt Mask (OTGIntMsk) **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0
1	modemismsk	Mode: Host and Device Mode Mismatch Interrupt Mask (ModeMisMsk) **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0

grxstsr

Receive Status Debug Read Register

Module Instance	Base Address	Register Address
i_usbotg_0_globgrp	0xFFB00000	0xFFB0001C
i_usbotg_1_globgrp	0xFFB40000	0xFFB4001C

Offset: 0x1C

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							fn RO 0x0				pktsts RO 0x0				dpid RO 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
dpid RO 0x0	bcnt RO 0x0											chnum RO 0x0			

grxstsr Fields

Bit	Name	Description	Access	Reset
24:21	fn	Mode: Device only Frame Number (FN) This is the least significant 4 bits of the (micro)Frame number in which the packet is received on the USB. This field is supported only when isochronous OUT endpoints are supported.	RO	0x0
20:17	pktsts	Mode: Host only Packet Status (PktSts) Indicates the status of the received packet 4'b0010: IN data packet received 4'b0011: IN transfer completed (triggers an interrupt) 4'b0101: Data toggle error (triggers an interrupt) 4'b0111: Channel halted (triggers an interrupt) Others: Reserved Mode: Device only Packet Status (PktSts) Indicates the status of the received packet 4'b0001: Global OUT NAK (triggers an interrupt) 4'b0010: OUT data packet received 4'b0011: OUT transfer completed (triggers an interrupt) 4'b0100: SETUP transaction completed (triggers an interrupt) 4'b0110: SETUP data packet received Others: Reserved **Value** **Description** 0x2 INDPRX 0x3 INTRCOM 0x5 DTTOG 0x7 CHHALT	RO	0x0

Bit	Name	Description	Access	Reset
16:15	dpid	Mode: Host only Data PID (DPID) Indicates the Data PID of the received packet 2'b00: DATA0 2'b10: DATA1 2'b01: DATA2 2'b11: MDATA Mode: Device only Data PID (DPID) Indicates the Data PID of the received OUT data packet 2'b00: DATA0 2'b10: DATA1 2'b01: DATA2 2'b11: MDATA **Value** **Description** 0x0 DATA0 0x1 DATA2 0x2 DATA1 0x3 MDATA	RO	0x0
14:4	bcnt	Mode: Host only Byte Count (BCnt) Indicates the byte count of the received IN data packet. Mode: Device only Byte Count (BCnt) Indicates the byte count of the received data packet.	RO	0x0

Bit	Name	Description	Access	Reset
3:0	chnum	Mode: Host only Channel Number (ChNum) Indicates the channel number to which the current received packet belongs. Mode: Device only Endpoint Number (EPNum) Indicates the endpoint number to which the current received packet belongs.	RO	0x0

grxstsp

Receive Status Read /Pop Register

Module Instance	Base Address	Register Address
i_usbotg_0_globgrp	0xFFB00000	0xFFB00020
i_usbotg_1_globgrp	0xFFB40000	0xFFB40020

Offset: 0x20

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							fn RO 0x0				pktsts RO 0x0				dpid RO 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
dpid RO 0x0	bcnt RO 0x0											chnum RO 0x0			

grxstsp Fields

Bit	Name	Description	Access	Reset
24:21	fn	Mode: Device only Frame Number (FN) This is the least significant 4 bits of the (micro)Frame number in which the packet is received on the USB. This field is supported only when isochronous OUT endpoints are supported.	RO	0x0
20:17	pktsts	Mode: Host only Packet Status (PktSts) Indicates the status of the received packet 4'b0010: IN data packet received 4'b0011: IN transfer completed (triggers an interrupt) 4'b0101: Data toggle error (triggers an interrupt) 4'b0111: Channel halted (triggers an interrupt) Others: Reserved Mode: Device only Packet Status (PktSts) Indicates the status of the received packet 4'b0001: Global OUT NAK (triggers an interrupt) 4'b0010: OUT data packet received 4'b0011: OUT transfer completed (triggers an interrupt) 4'b0100: SETUP transaction completed (triggers an interrupt) 4'b0110: SETUP data packet received Others: Reserved **Value** — **Description** 0x0 — DATA0 0x1 — DATA2 0x2 — DATA1 0x3 — MDATA	RO	0x0

Bit	Name	Description	Access	Reset
16:15	dpid	Mode: Host only Data PID (DPID) Indicates the Data PID of the received packet 2'b00: DATA0 2'b10: DATA1 2'b01: DATA2 2'b11: MDATA Mode: Device only Data PID (DPID) Indicates the Data PID of the received OUT data packet 2'b00: DATA0 2'b10: DATA1 2'b01: DATA2 2'b11: MDATA **Value** **Description** 0x0 DATA0 0x1 DATA2 0x2 DATA1 0x3 MDATA	RO	0x0
14:4	bcnt	Mode: Host only Byte Count (BCnt) Indicates the byte count of the received IN data packet. Mode: Device only Byte Count (BCnt) Indicates the byte count of the received data packet.	RO	0x0

Bit	Name	Description	Access	Reset
3:0	chnum	Mode: Host only Channel Number (ChNum) Indicates the channel number to which the current received packet belongs. Mode: Device only Endpoint Number (EPNum) Indicates the endpoint number to which the current received packet belongs.	RO	0x0

grxfsiz

Receive FIFO Size Register

Module Instance	Base Address	Register Address
i_usbotg_0_globgrp	0xFFB00000	0xFFB00024
i_usbotg_1_globgrp	0xFFB40000	0xFFB40024

Offset: 0x24

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		rxfdep RW 0x2000													

grxfsiz Fields

Bit	Name	Description	Access	Reset
13:0	rxfdep	Mode: Host and Device RxFIFO Depth (RxFDep) This value is in terms of 32-bit words. Minimum value is 16 Maximum value is 32,768 The power-on reset value of this register is specified as the Largest Rx Data FIFO Depth during configuration. If Enable Dynamic FIFO Sizing was selected, you can write a new value in this field. Programmed values must not exceed the power-on value	RW	0x2000

gnptxfsiz

Non-periodic Transmit FIFO Size Register

Module Instance	Base Address	Register Address
i_usbotg_0_globgrp	0xFFB00000	0xFFB00028
i_usbotg_1_globgrp	0xFFB40000	0xFFB40028

Offset: 0x28

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
nptxfdep RW 0x2000															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
nptxfstaddr RW 0x2000															

gnptxfsiz Fields

Bit	Name	Description	Access	Reset
31:16	nptxfdep	Mode: Host only Non-periodic TxFIFO Depth (NPTxFDep) For host mode, this field is always valid. For Device mode, this field is valid for shared fifo This value is in terms of 32-bit words. Minimum value is 16 Maximum value is 32,768 Programmed values must not exceed the power-on value. Mode: Device only IN Endpoint TxFIFO 0 Depth (INEPTxF0Dep) This value is in terms of 32-bit words. Minimum value is 16 Maximum value is 32,768 This field is determined by Enable Dynamic FIFO Sizing Programmed values must not exceed the power-on value.	RW	0x2000
15:0	nptxfstaddr	Mode: Host only Non-periodic Transmit RAM Start Address (NPTxFStAddr) For host mode, this field is always valid. This field contains the memory start address For Non-periodic Transmit FIFO RAM. Programmed values must not exceed the power-on value. Mode: Device only IN Endpoint FIFO0 Transmit RAM Start Address (INEPTxF0StAddr) This field contains the memory start address For IN Endpoint Transmit FIFO# 0. Programmed values must not exceed the power-on value.	RW	0x2000

gnptxsts

Non-periodic Transmit FIFO/Queue Status Register

Module Instance	Base Address	Register Address
i_usbotg_0_globgrp	0xFFB00000	0xFFB0002C
i_usbotg_1_globgrp	0xFFB40000	0xFFB4002C

Offset: 0x2C

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	nptxqtop RO 0x0							nptxqspcavail RO 0x8							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
nptxfspcavail RO 0x2000															

gnptxsts Fields

Bit	Name	Description	Access	Reset
30:24	nptxqtop	Top of the Non-periodic Transmit Request Queue (NPTxQTop) Entry in the Non-periodic Tx Request Queue that is currently being processed by the MAC. Bits [30:27]: Channel/endpoint number Bits [26:25]: - 2'b00: IN/OUT token - 2'b01: Zero-length transmit packet (device IN/host OUT) - 2'b10: PING/CSPLIT token - 2'b11: Channel halt command Bit [24]: Terminate (last Entry For selected channel/endpoint) **Value** — **Description** 0x0 — INOUTTK 0x1 — ZEROTX 0x2 — PINGCSPLIT 0x3 — CHNHALT	RO	0x0

Bit	Name	Description	Access	Reset
23:16	nptxqspcavail	Non-periodic Transmit Request Queue Space Available (NPTxQSpcAvail) Indicates the amount of free space available in the Non-periodic Transmit Request Queue. This queue holds both IN and OUT requests in Host mode. Device mode has only IN requests. 8'h0: Non-periodic Transmit Request Queue is full 8'h1: 1 location available 8'h2: 2 locations available n: n locations available (0 <= n <= 8) Others: Reserved **Value** — **Description** 0x0 — FULL 0x1 — QUE1 0x2 — QUE2 0x3 — QUE3 0x4 — QUE4 0x5 — QUE5 0x6 — QUE6 0x7 — QUE7 0x8 — QUE8	RO	0x8
15:0	nptxfspcavail	Non-periodic TxFIFO Space Avail (NPTxFSpcAvail) Indicates the amount of free space available in the Non-periodic TxFIFO. Values are in terms of 32-bit words. 16'h0: Non-periodic TxFIFO is full 16'h1: 1 word available 16'h2: 2 words available 16'hn: n words available (where 0 <= n <= 32,768) 16'h8000: 32,768 words available Others: Reserved	RO	0x2000

gpvndctl

PHY Vendor Control Register

Module Instance	Base Address	Register Address
i_usbotg_0_globgrp	0xFFB00000	0xFFB00034
i_usbotg_1_globgrp	0xFFB40000	0xFFB40034

Offset: 0x34

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
disulpidrvr RW 0x0	Reserved			vstsdone RW 0x0	vstsbsy RO 0x0	newregreq RW 0x0	Reserved		regwr RW 0x0	regaddr RW 0x0					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
vctrl RW 0x0								regdata RW 0x0							

gpvndctl Fields

Bit	Name	Description	Access	Reset
31	disulpidrvr	Disable ULPI Drivers (DisUlpiDrvr) The application sets this bit when it has finished processing the ULPI Carkit Interrupt (GINTSTS.ULPICKINT). When Set, the DWC_otg core disables drivers For output signals and masks input signal For the ULPI interface. DWC_otg clears this bit before enabling the ULPI interface. **Value** — **Description** 0x0 — ENABLED 0x1 — DISABLED	RW	0x0
27	vstsdone	VStatus Done (VStsDone) The core sets this bit when the vendor control access is done. This bit is cleared by the core when the application sets the New Register Request bit (bit 25). **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
26	vstsbsy	VStatus Busy (VStsBsy) The core sets this bit when the vendor control access is in progress and clears this bit when done. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RO	0x0

Bit	Name	Description	Access	Reset
25	newregreq	New Register Request (NewRegReq) The application sets this bit For a new vendor control access. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
22	regwr	Register Write (RegWr) Set this bit For register writes, and clear it For register reads. **Value** — **Description** 0x0 — READ 0x1 — WRITE	RW	0x0
21:16	regaddr	Register Address (RegAddr) The 6-bit PHY register address For immediate PHY Register Set access. Set to 6'h2F For Extended PHY Register Set access.	RW	0x0
15:8	vctrl	UTMI+ Vendor Control Register Address (VCtrl) The 4-bit register address a vendor defined 4-bit parallel output bus. Bits 11:8 of this field are placed on utmi_vcontrol[3:0]. ULPI Extended Register Address (ExtRegAddr) The 6-bit PHY extended register address.	RW	0x0
7:0	regdata	Register Data (RegData) Contains the write data For register write. Read data For register read, valid when VStatus Done is Set.	RW	0x0

ggpio

General Purpose Input/Output Register

Module Instance	Base Address	Register Address
i_usbotg_0_globgrp	0xFFB00000	0xFFB00038
i_usbotg_1_globgrp	0xFFB40000	0xFFB40038

Offset: 0x38

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gpo RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
gpi RO 0x0															

ggpio Fields

Bit	Name	Description	Access	Reset
31:16	gpo	General Purpose Output (GPO) This field is driven as an output from the core, gp_o[15:0]. The application can program this field to determine the corresponding value on the gp_o[15:0] output.	RW	0x0
15:0	gpi	General Purpose Input (GPI) This field's read value reflects the gp_i[15:0] core input value.	RO	0x0

guid

User ID Register

Module Instance	Base Address	Register Address
i_usbotg_0_globgrp	0xFFB00000	0xFFB0003C
i_usbotg_1_globgrp	0xFFB40000	0xFFB4003C

Offset: 0x3C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
guid RO 0x12345678															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
guid RO 0x12345678															

guid Fields

Bit	Name	Description	Access	Reset
31:0	guid	User ID (UserID) Application-programmable ID field. Reset: Configurable	RO	0x12345678

gsnpsid

Synopsys ID Register

Module Instance	Base Address	Register Address
i_usbotg_0_globgrp	0xFFB00000	0xFFB00040
i_usbotg_1_globgrp	0xFFB40000	0xFFB40040

Offset: 0x40

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gsnpsid RO 0x4F54320A															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
gsnpsid RO 0x4F54320A															

gsnpsid Fields

Bit	Name	Description	Access	Reset
31:0	gsnpsid	Release number of the DWC_otg core being used is currently OTG	RO	0x4F54320A

ghwcfg1

User HW Config1 Register

Module Instance	Base Address	Register Address
i_usbotg_0_globgrp	0xFFB00000	0xFFB00044
i_usbotg_1_globgrp	0xFFB40000	0xFFB40044

Offset: 0x44

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ghwcfg1 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ghwcfg1 RO 0x0															

ghwcfg1 Fields

Bit	Name	Description	Access	Reset
31:0	ghwcfg1	This 32-bit field uses two bits per endpoint to determine the endpoint direction. Endpoint Bits [31:30]: Endpoint 15 direction Bits [29:28]: Endpoint 14 direction ... Bits [3:2]: Endpoint 1 direction Bits[1:0]: Endpoint 0 direction (always BIDIR) Direction 2'b00: BIDIR (IN and OUT) endpoint 2'b01: IN endpoint 2'b10: OUT endpoint 2'b11: Reserved **Value** **Description** 0x0 BDIR 0x1 INENDPT 0x2 OUTENDPT 0x3 RESERVED	RO	0x0

ghwcfg2

User HW Config2 Register

Module Instance	Base Address	Register Address
i_usbotg_0_globgrp	0xFFB00000	0xFFB00048
i_usbotg_1_globgrp	0xFFB40000	0xFFB40048

Offset: 0x48

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	tknqdepth RO 0x8					ptxqdepth RO 0x0		nptxqdepth RO 0x2		Reserved	multiprocintrpt RO 0x0	dynfifosizing RO 0x1	periosupport RO 0x1	numhstchnl RO 0xF	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
numhstchnl RO 0xF		numdeveps RO 0xF				fsphytype RO 0x0		hsphytype RO 0x2		singpnt RO 0x0	otgarch RO 0x2		otgmode RO 0x0		

ghwcfg2 Fields

Bit	Name	Description	Access	Reset
30:26	tknqdepth	Device Mode IN Token Sequence Learning Queue Depth (TknQDepth) Range: 0-30	RO	0x8
25:24	ptxqdepth	Host Mode Periodic Request Queue Depth (PTxQDepth) 2'b00: 2 2'b01: 4 2'b10: 8 2'b11:16 **Value** **Description** 0x0 QUE2 0x1 QUE4 0x2 QUE8 0x3 QUE16	RO	0x0

Bit	Name	Description	Access	Reset
23:22	nptxqdepth	Non-periodic Request Queue Depth (NPTxQDepth) 2'b00: 2 2'b01: 4 2'b10: 8 Others: Reserved **Value** **Description** 0x0 TWO 0x1 FOUR 0x2 EIGHT	RO	0x2
20	multiprocintrpt	Multi Processor Interrupt Enabled (MultiProcIntrpt) 1'b0: No 1'b1: Yes **Value** **Description** 0x0 DISABLED	RO	0x0
19	dynfifosizing	Dynamic FIFO Sizing Enabled (DynFifoSizing) 1'b0: No 1'b1: Yes **Value** **Description** 0x1 ENABLED	RO	0x1
18	periosupport	Periodic OUT Channels Supported in Host Mode (PerioSupport) 1'b0: No 1'b1: Yes **Value** **Description** 0x1 ENABLED	RO	0x1

Bit	Name	Description	Access	Reset
17:14	numhstchnl	Number of Host Channels (NumHstChnl) Indicates the number of host channels supported by the core in Host mode. The range of this field is 0-15: 0 specifies 1 channel, 15 specifies 16 channels. **Value** **Description** 0xa HOSTCH10 0xb HOSTCH11 0xc HOSTCH12 0xd HOSTCH13 0xe HOSTCH14 0xf HOSTCH15 0x0 HOSTCH0 0x1 HOSTCH1 0x2 HOSTCH2 0x3 HOSTCH3 0x4 HOSTCH4 0x5 HOSTCH5 0x6 HOSTCH6 0x7 HOSTCH7 0x8 HOSTCH8 0x9 HOSTCH9	RO	0xF

Bit	Name	Description	Access	Reset
13:10	numdeveps	Number of Device Endpoints (NumDevEps) Indicates the number of device endpoints supported by the core in Device mode in addition to control endpoint 0. The range of this field is 1-15. Value: Description 0xa: ENDPT10 0xb: ENDPT11 0xc: ENDPT12 0xd: ENDPT13 0xe: ENDPT14 0xf: ENDPT15 0x0: ENDPT0 0x1: ENDPT1 0x2: ENDPT2 0x3: ENDPT3 0x4: ENDPT4 0x5: ENDPT5 0x6: ENDPT6 0x7: ENDPT7 0x8: ENDPT8 0x9: ENDPT9	RO	0xF
9:8	fsphytype	Full-Speed PHY Interface Type (FSPhyType) 2'b00: Full-speed interface not supported 2'b01: Dedicated full-speed interface 2'b10: FS pins shared with UTMI+ pins 2'b11: FS pins shared with ULPI pins Value: Description 0x2: FULLSPEED	RO	0x0

Bit	Name	Description	Access	Reset
7:6	hsphytype	High-Speed PHY Interface Type (HSPhyType) 2'b00: High-Speed interface not supported 2'b01: UTMI+ 2'b10: ULPI 2'b11: UTMI+ and ULPI **Value** **Description** 0x0 NOHS 0x2 ULPI	RO	0x2
5	singpnt	Point-to-Point (SingPnt) 1'b0: Multi-point application (hub and split support) 1'b1: Single-point application (no hub and split support) **Value** **Description** 0x1 SINGLEPOINT	RO	0x0
4:3	otgarch	Architecture (OtgArch) 2'b00: Slave-Only 2'b01: External DMA 2'b10: Internal DMA Others: Reserved **Value** **Description** 0x2 DMAMODE	RO	0x2

Bit	Name	Description	Access	Reset
2:0	otgmode	Mode of Operation (OtgMode) 3'b000: HNP- and SRP-Capable OTG (Host & Device) 3'b001: SRP-Capable OTG (Host & Device) 3'b010: Non-HNP and Non-SRP Capable OTG (Host & Device) 3'b011: SRP-Capable Device 3'b100: Non-OTG Device 3'b101: SRP-Capable Host 3'b110: Non-OTG Host Others: Reserved **Value** — **Description** 0x0 — HNPSRP 0x1 — SRPOTG 0x2 — NHNPNSRP 0x3 — SRPCAPD 0x4 — NONOTGD 0x5 — SRPCAPH 0x6 — NONOTGH	RO	0x0

ghwcfg3

User HW Config3 Register

Module Instance	Base Address	Register Address
i_usbotg_0_globgrp	0xFFB00000	0xFFB0004C
i_usbotg_1_globgrp	0xFFB40000	0xFFB4004C

Offset: 0x4C

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dfifodepth RO 0x1F80															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
lpmmode RO 0x0	bcsupport RO 0x0	hsicmode RO 0x0	adpsupport RO 0x0	rsttype RO 0x0	optfeature RO 0x0	vndctlsupt RO 0x1	i2cintsel RO 0x0	otgen RO 0x1	pktsizewidth RO 0x6			xfersizewidth RO 0x8			

ghwcfg3 Fields

Bit	Name	Description	Access	Reset
31:16	dfifodepth	DFIFO Depth (DfifoDepth - EP_LOC_CNT) This value is in terms of 32-bit words. Minimum value is 32 Maximum value is 32,768	RO	0x1F80
15	lpmmode	LPM mode specified for Mode of Operation. **Value** **Description** 0x0 DISABLED	RO	0x0
14	bcsupport	This bit indicates the HS OTG controller support for Battery Charger. 0 - No Battery Charger Support 1 - Battery Charger support present. **Value** **Description** 0x0 DISABLED	RO	0x0

Bit	Name	Description	Access	Reset
13	hsicmode	HSIC mode specified for Mode of Operation Value Range: 0 - 1 1: HSIC-capable with shared UTMI PHY interface 0: Non-HSIC-capable **Value** **Description** 0x0 DISABLED	RO	0x0
12	adpsupport	This bit indicates whether ADP logic is present within or external to the HS OTG controller 0: No ADP logic present with HSOTG controller 1: ADP logic is present along with HSOTG controller. **Value** **Description** 0x1 ENABLED	RO	0x0
11	rsttype	Reset Style For Clocked always Blocks in RTL (RstType) 1'b0: Asynchronous reset is used in the core 1'b1: Synchronous reset is used in the core **Value** **Description** 0x0 ENABLED	RO	0x0

Bit	Name	Description	Access	Reset
10	optfeature	Optional Features Removed (OptFeature) Indicates whether the User ID register, GPIO interface ports, and SOF toggle and counter ports were removed For gate count optimization by enabling Remove Optional Features. 1'b0: No 1'b1: Yes **Value** **Description** 0x0 DISABLED	RO	0x0
9	vndctlsupt	Vendor Control Interface Support (VndctlSupt) 1'b0: Vendor Control Interface is not available on the core. 1'b1: Vendor Control Interface is available. **Value** **Description** 0x1 ENABLED	RO	0x1
8	i2cintsel	I2C Selection (I2CIntSel) 1'b0: I2C Interface is not available on the core. 1'b1: I2C Interface is available on the core. **Value** **Description** 0x0 DISABLED	RO	0x0
7	otgen	OTG Function Enabled (OtgEn) The application uses this bit to indicate the DWC_otg core's OTG capabilities. 1'b0: Not OTG capable 1'b1: OTG Capable **Value** **Description** 0x1 ENABLED	RO	0x1

Bit	Name	Description	Access	Reset
6:4	pktsizewidth	Width of Packet Size Counters (PktSizeWidth) 3'b000: 4 bits 3'b001: 5 bits 3'b010: 6 bits 3'b011: 7 bits 3'b100: 8 bits 3'b101: 9 bits 3'b110: 10 bits Others: Reserved Value — Description 0x0 — BITS4 0x1 — BITS5 0x2 — BITS6 0x3 — BITS7 0x4 — BITS8 0x5 — BITS9 0x6 — BITS10	RO	0x6
3:0	xfersizewidth	Width of Transfer Size Counters (XferSizeWidth) 4'b0000: 11 bits 4'b0001: 12 bits ... 4'b1000: 19 bits Others: Reserved Value — Description 0x0 — WIDTH11 0x1 — WIDTH12 0x2 — WIDTH13 0x3 — WIDTH14 0x4 — WIDTH15 0x5 — WIDTH16 0x6 — WIDTH17 0x7 — WIDTH18 0x8 — WIDTH19	RO	0x8

ghwcfg4

User HW Config4 Register

Module Instance	Base Address	Register Address
i_usbotg_0_globgrp	0xFFB00000	0xFFB00050
i_usbotg_1_globgrp	0xFFB40000	0xFFB40050

Offset: 0x50

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dma RO 0x1	dma_configuration RO 0x1	ineps RO 0xF				dedfifomode RO 0x1	sessendflt r RO 0x0	bvalidfltr RO 0x0	avalidfltr RO 0x0	vbusvalidfltr RO 0x0	iddgfltr RO 0x0	numctleps RO 0xF			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
phydatawidth RO 0x0		Reserved						extendedhibernation RO 0x0	hibernation RO 0x0	ahbfreq RO 0x1	partialpwrdn RO 0x0	numdevperioeps RO 0x0			

ghwcfg4 Fields

Bit	Name	Description	Access	Reset
31	dma	Scatter/Gather DMA configuration 1'b0: Non Dynamic configuration 1'b1: Dynamic configuration **Value** **Description** 0x1 ENABLED	RO	0x1

Bit	Name	Description	Access	Reset
30	dma_configuration	Scatter/Gather DMA configuration 1'b0: Non-Scatter/Gather DMA configuration 1'b1: Scatter/Gather DMA configuration **Value** **Description** 0x0 NONSCATTER 0x1 SCATTER	RO	0x1
29:26	ineps	Number of Device Mode IN Endpoints Including Control Endpoints (INEps) Range 0 -15 0 : 1 IN Endpoint 1 : 2 IN Endpoints .... 15 : 16 IN Endpoints **Value** **Description** 0xa ENDPT11 0xb ENDPT12 0xc ENDPT13 0xd ENDPT14 0xe ENDPT15 0xf ENDPT16 0x0 ENDPT1 0x1 ENDPT2 0x2 ENDPT3 0x3 ENDPT4 0x4 ENDPT5 0x5 ENDPT6 0x6 ENDPT7 0x7 ENDPT8 0x8 ENDPT9 0x9 ENDPT10	RO	0xF

Bit	Name	Description	Access	Reset
25	dedfifomode	Enable Dedicated Transmit FIFO For device IN Endpoints (DedFifoMode) 1'b0 : Dedicated Transmit FIFO Operation not enabled. 1'b1 : Dedicated Transmit FIFO Operation enabled. **Value** **Description** 0x1 ENABLED	RO	0x1
24	sessendfltr	session_end Filter Enabled (SessEndFltr) 1'b0: No filter 1'b1: Filter **Value** **Description** 0x0 DISABLED	RO	0x0
23	bvalidfltr	b_valid Filter Enabled (BValidFltr) 1'b0: No filter 1'b1: Filter **Value** **Description** 0x0 DISABLED	RO	0x0
22	avalidfltr	a_valid Filter Enabled (AValidFltr) 1'b0: No filter 1'b1: Filter **Value** **Description** 0x0 DISABLED	RO	0x0
21	vbusvalidfltr	VBUS Valid Filter Enabled (VBusValidFltr) 1'b0: No filter 1'b1: Filter **Value** **Description** 0x0 DISABLED	RO	0x0

Bit	Name	Description	Access	Reset
20	iddgfltr	IDDIG Filter Enable (IddgFltr) 1'b0: No filter 1'b1: Filter **Value** **Description** 0x0 DISABLED	RO	0x0
19:16	numctleps	Number of Device Mode Control Endpoints in Addition to Endpoint 0 (NumCtlEps) Range: 0-15 **Value** **Description** 0xa ENDPT10 0xb ENDPT11 0xc ENDPT12 0xd ENDPT13 0xe ENDPT14 0xf ENDPT15 0x0 ENDPT0 0x1 ENDPT1 0x2 ENDPT2 0x3 ENDPT3 0x4 ENDPT4 0x5 ENDPT5 0x6 ENDPT6 0x7 ENDPT7 0x8 ENDPT8 0x9 ENDPT9	RO	0xF

Bit	Name	Description	Access	Reset
15:14	phydatawidth	UTMI+ PHY/ULPI-to-Internal UTMI+ Wrapper Data Width (PhyDataWidth) When a ULPI PHY is used, an internal wrapper converts ULPI to UTMI+ . 2'b00: 8 bits 2'b01: 16 bits 2'b10: 8/16 bits, software selectable Others: Reserved	RO	0x0
7	extendedhibernation	Enable Hibernation 1'b0: Extended Hibernation feature not enabled 1'b1: Extended Hibernation feature enabled	RO	0x0
6	hibernation	Enable Hibernation (Hibernation) 1'b0: Hibernation feature not enabled 1'b1: Hibernation feature enabled **Value** **Description** 0x0 DISABLED	RO	0x0
5	ahbfreq	Minimum AHB Frequency Less Than 60 MHz (AhbFreq) 1'b0: No 1'b1: Yes **Value** **Description** 0x1 ENABLED	RO	0x1
4	partialpwrdn	Enable Partial Power Down (PartialPwrDn) 1'b0: Partial Power Down Not Enabled 1'b1: Partial Power Down Enabled **Value** **Description** 0x0 DISABLED	RO	0x0

Bit	Name	Description	Access	Reset
3:0	numdevperioeps	Number of Device Mode Periodic IN Endpoints (NumDevPerioEps) Range: 0-15	RO	0x0

gdfifocfg

Global DFIFO Configuration Register

Module Instance	Base Address	Register Address
i_usbotg_0_globgrp	0xFFB00000	0xFFB0005C
i_usbotg_1_globgrp	0xFFB40000	0xFFB4005C

Offset: 0x5C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
epinfobaseaddr RW 0x1F80															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
gdfifocfg RW 0x2000															

gdfifocfg Fields

Bit	Name	Description	Access	Reset
31:16	epinfobaseaddr	EPInfoBaseAddr This field provides the start address of the EP info controller.	RW	0x1F80

Bit	Name	Description	Access	Reset
15:0	gdfifocfg	GDFIFOCfg This field is for dynamic programming of the DFIFO Size. This value takes effect only when the application programs a non zero value to this register. The value programmed must conform to the guidelines described in 'FIFO RAM Allocation'. The DWC_otg core does not have any corrective logic if the FIFO sizes are programmed incorrectly.	RW	0x2000

hptxfsiz

Host Periodic Transmit FIFO Size Register

Module Instance	Base Address	Register Address
i_usbotg_0_globgrp	0xFFB00000	0xFFB00100
i_usbotg_1_globgrp	0xFFB40000	0xFFB40100

Offset: 0x100

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		ptxfsize RW 0x2000													
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	ptxfstaddr RW 0x4000														

hptxfsiz Fields

Bit	Name	Description	Access	Reset
29:16	ptxfsize	Host Periodic TxFIFO Depth (PTxFSize) This value is in terms of 32-bit words. Minimum value is 16 Maximum value is 32,768 The power-on reset value of this register is specified as the Largest Host Mode Periodic Tx Data FIFO Depth. Programmed values must not exceed the power-on value.	RW	0x2000
14:0	ptxfstaddr	Host Periodic TxFIFO Start Address (PTxFStAddr) The power-on reset value of this register is the sum of the Largest Rx Data FIFO Depth and Largest Non-periodic Tx Data FIFO Depth Programmed values must not exceed the power-on value	RW	0x4000

dieptxf1

Device IN Endpoint Transmit FIFO Size Register 1

Module Instance	Base Address	Register Address
i_usbotg_0_globgrp	0xFFB00000	0xFFB00104
i_usbotg_1_globgrp	0xFFB40000	0xFFB40104

Offset: 0x104

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		inepntxfdep RW 0x2000													
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		inepntxfstaddr RW 0x4000													

dieptxf1 Fields

Bit	Name	Description	Access	Reset
29:16	inepntxfdep	IN Endpoint TxFIFO Depth (INEPnTxFDep) This value is in terms of 32-bit words. Minimum value is 16 Maximum value is 32,768 The power-on reset value of this register is specified as the Largest IN Endpoint FIFO number Depth Programmed values must not exceed the power-on value	RW	0x2000
14:0	inepntxfstaddr	IN Endpoint FIFOn Transmit RAM Start Address (INEPnTxFStAddr) This field contains the memory start address For IN endpoint Transmit FIFOn (0<n< = 15). The power-on reset value of this register is specified as the Largest Rx Data FIFO Depth Programmed values must not exceed the power-on value.	RW	0x4000

dieptxf2

Device IN Endpoint Transmit FIFO Size Register 2

Module Instance	Base Address	Register Address
i_usbotg_0_globgrp	0xFFB00000	0xFFB00108

Module Instance	Base Address	Register Address
i_usbotg_1_globgrp	0xFFB40000	0xFFB40108

Offset: 0x108

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		inepntxfdep RW 0x2000													
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	inepntxfstaddr RW 0x6000														

dieptxf2 Fields

Bit	Name	Description	Access	Reset
29:16	inepntxfdep	IN Endpoint TxFIFO Depth (INEPnTxFDep) This value is in terms of 32-bit words. Minimum value is 16 Maximum value is 32,768 The power-on reset value of this register is specified as the Largest IN Endpoint FIFO number Depth Programmed values must not exceed the power-on value	RW	0x2000

Bit	Name	Description	Access	Reset
14:0	inepntxfstaddr	IN Endpoint FIFOn Transmit RAM Start Address (INEPnTxFStAddr) This field contains the memory start address For IN endpoint Transmit FIFOn (0<n< = 15). The power-on reset value of this register is specified as the Largest Rx Data FIFO Depth Programmed values must not exceed the power-on value.	RW	0x6000

dieptxf3

Device IN Endpoint Transmit FIFO Size Register 3

Module Instance	Base Address	Register Address
i_usbotg_0_globgrp	0xFFB00000	0xFFB0010C
i_usbotg_1_globgrp	0xFFB40000	0xFFB4010C

Offset: 0x10C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		inepntxfdep RW 0x2000													
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
inepntxfstaddr RW 0x8000															

dieptxf3 Fields

Bit	Name	Description	Access	Reset
29:16	inepntxfdep	IN Endpoint TxFIFO Depth (INEPnTxFDep) This value is in terms of 32-bit words. Minimum value is 16 Maximum value is 32,768 The power-on reset value of this register is specified as the Largest IN Endpoint FIFO number Depth Programmed values must not exceed the power-on value	RW	0x2000
15:0	inepntxfstaddr	IN Endpoint FIFOn Transmit RAM Start Address (INEPnTxFStAddr) This field contains the memory start address For IN endpoint Transmit FIFOn (0<n< = 15). The power-on reset value of this register is specified as the Largest Rx Data FIFO Depth Programmed values must not exceed the power-on value.	RW	0x8000

dieptxf4

Device IN Endpoint Transmit FIFO Size Register 4

Module Instance	Base Address	Register Address
i_usbotg_0_globgrp	0xFFB00000	0xFFB00110
i_usbotg_1_globgrp	0xFFB40000	0xFFB40110

Offset: 0x110

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		inepntxfdep RW 0x2000													
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
inepntxfstaddr RW 0xA000															

dieptxf4 Fields

Bit	Name	Description	Access	Reset
29:16	inepntxfdep	IN Endpoint TxFIFO Depth (INEPnTxFDep) This value is in terms of 32-bit words. Minimum value is 16 Maximum value is 32,768 The power-on reset value of this register is specified as the Largest IN Endpoint FIFO number Depth Programmed values must not exceed the power-on value	RW	0x2000
15:0	inepntxfstaddr	IN Endpoint FIFOn Transmit RAM Start Address (INEPnTxFStAddr) This field contains the memory start address For IN endpoint Transmit FIFOn (0<n< = 15). The power-on reset value of this register is specified as the Largest Rx Data FIFO Depth Programmed values must not exceed the power-on value.	RW	0xA000

dieptxf5

Device IN Endpoint Transmit FIFO Size Register 5

Module Instance	Base Address	Register Address
i_usbotg_0_globgrp	0xFFB00000	0xFFB00114

Module Instance	Base Address	Register Address
i_usbotg_1_globgrp	0xFFB40000	0xFFB40114

Offset: 0x114

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		inepntxfdep RW 0x2000													
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
inepntxfstaddr RW 0xC000															

dieptxf5 Fields

Bit	Name	Description	Access	Reset
29:16	inepntxfdep	IN Endpoint TxFIFO Depth (INEPnTxFDep) This value is in terms of 32-bit words. Minimum value is 16 Maximum value is 32,768 The power-on reset value of this register is specified as the Largest IN Endpoint FIFO number Depth Programmed values must not exceed the power-on value	RW	0x2000

Bit	Name	Description	Access	Reset
15:0	inepntxfstaddr	IN Endpoint FIFOn Transmit RAM Start Address (INEPnTxFStAddr) This field contains the memory start address For IN endpoint Transmit FIFOn (0<n< = 15). The power-on reset value of this register is specified as the Largest Rx Data FIFO Depth Programmed values must not exceed the power-on value.	RW	0xC000

dieptxf6

Device IN Endpoint Transmit FIFO Size Register 6

Module Instance	Base Address	Register Address
i_usbotg_0_globgrp	0xFFB00000	0xFFB00118
i_usbotg_1_globgrp	0xFFB40000	0xFFB40118

Offset: 0x118

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		inepntxfdep RW 0x2000													
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
inepntxfstaddr RW 0xE000															

dieptxf6 Fields

Bit	Name	Description	Access	Reset
29:16	inepntxfdep	IN Endpoint TxFIFO Depth (INEPnTxFDep) This value is in terms of 32-bit words. Minimum value is 16 Maximum value is 32,768 The power-on reset value of this register is specified as the Largest IN Endpoint FIFO number Depth Programmed values must not exceed the power-on value	RW	0x2000
15:0	inepntxfstaddr	IN Endpoint FIFOn Transmit RAM Start Address (INEPnTxFStAddr) This field contains the memory start address For IN endpoint Transmit FIFOn (0<n< = 15). The power-on reset value of this register is specified as the Largest Rx Data FIFO Depth Programmed values must not exceed the power-on value.	RW	0xE000

dieptxf7

Device IN Endpoint Transmit FIFO Size Register 7

Module Instance	Base Address	Register Address
i_usbotg_0_globgrp	0xFFB00000	0xFFB0011C
i_usbotg_1_globgrp	0xFFB40000	0xFFB4011C

Offset: 0x11C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		inepntxfdep RW 0x2000													
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
inepntxfstaddr RW 0x0															

dieptxf7 Fields

Bit	Name	Description	Access	Reset
29:16	inepntxfdep	IN Endpoint TxFIFO Depth (INEPnTxFDep) This value is in terms of 32-bit words. Minimum value is 16 Maximum value is 32,768 The power-on reset value of this register is specified as the Largest IN Endpoint FIFO number Depth Programmed values must not exceed the power-on value	RW	0x2000
15:0	inepntxfstaddr	IN Endpoint FIFOn Transmit RAM Start Address (INEPnTxFStAddr) This field contains the memory start address For IN endpoint Transmit FIFOn (0<n< = 15). The power-on reset value of this register is specified as the Largest Rx Data FIFO Depth Programmed values must not exceed the power-on value.	RW	0x0

dieptxf8

Device IN Endpoint Transmit FIFO Size Register 8

Module Instance	Base Address	Register Address
i_usbotg_0_globgrp	0xFFB00000	0xFFB00120

Module Instance	Base Address	Register Address
i_usbotg_1_globgrp	0xFFB40000	0xFFB40120

Offset: 0x120

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		inepntxfdep RW 0x2000													
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
inepntxfstaddr RW 0x2000															

dieptxf8 Fields

Bit	Name	Description	Access	Reset
29:16	inepntxfdep	IN Endpoint TxFIFO Depth (INEPnTxFDep) This value is in terms of 32-bit words. Minimum value is 16 Maximum value is 32,768 The power-on reset value of this register is specified as the Largest IN Endpoint FIFO number Depth Programmed values must not exceed the power-on value	RW	0x2000

Bit	Name	Description	Access	Reset
15:0	inepntxfstaddr	IN Endpoint FIFOn Transmit RAM Start Address (INEPnTxFStAddr) This field contains the memory start address For IN endpoint Transmit FIFOn (0<n< = 15). The power-on reset value of this register is specified as the Largest Rx Data FIFO Depth Programmed values must not exceed the power-on value.	RW	0x2000

dieptxf9

Device IN Endpoint Transmit FIFO Size Register 9

Module Instance	Base Address	Register Address
i_usbotg_0_globgrp	0xFFB00000	0xFFB00124
i_usbotg_1_globgrp	0xFFB40000	0xFFB40124

Offset: 0x124

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		inepntxfdep RW 0x2000													
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
inepntxfstaddr RW 0x4000															

dieptxf9 Fields

Bit	Name	Description	Access	Reset
29:16	inepntxfdep	IN Endpoint TxFIFO Depth (INEPnTxFDep) This value is in terms of 32-bit words. Minimum value is 16 Maximum value is 32,768 The power-on reset value of this register is specified as the Largest IN Endpoint FIFO number Depth Programmed values must not exceed the power-on value	RW	0x2000
15:0	inepntxfstaddr	IN Endpoint FIFOn Transmit RAM Start Address (INEPnTxFStAddr) This field contains the memory start address For IN endpoint Transmit FIFOn (0<n< = 15). The power-on reset value of this register is specified as the Largest Rx Data FIFO Depth Programmed values must not exceed the power-on value.	RW	0x4000

dieptxf10

Device IN Endpoint Transmit FIFO Size Register 10

Module Instance	Base Address	Register Address
i_usbotg_0_globgrp	0xFFB00000	0xFFB00128
i_usbotg_1_globgrp	0xFFB40000	0xFFB40128

Offset: 0x128

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		inepntxfdep RW 0x2000													
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
inepntxfstaddr RW 0x6000															

dieptxf10 Fields

Bit	Name	Description	Access	Reset
29:16	inepntxfdep	IN Endpoint TxFIFO Depth (INEPnTxFDep) This value is in terms of 32-bit words. Minimum value is 16 Maximum value is 32,768 The power-on reset value of this register is specified as the Largest IN Endpoint FIFO number Depth Programmed values must not exceed the power-on value	RW	0x2000
15:0	inepntxfstaddr	IN Endpoint FIFOn Transmit RAM Start Address (INEPnTxFStAddr) This field contains the memory start address For IN endpoint Transmit FIFOn (0<n< = 15). The power-on reset value of this register is specified as the Largest Rx Data FIFO Depth Programmed values must not exceed the power-on value.	RW	0x6000

dieptxf11

Device IN Endpoint Transmit FIFO Size Register 11

Module Instance	Base Address	Register Address
i_usbotg_0_globgrp	0xFFB00000	0xFFB0012C

Module Instance	Base Address	Register Address
i_usbotg_1_globgrp	0xFFB40000	0xFFB4012C

Offset: 0x12C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		inepntxfdep RW 0x2000													
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
inepntxfstaddr RW 0x8000															

dieptxf11 Fields

Bit	Name	Description	Access	Reset
29:16	inepntxfdep	IN Endpoint TxFIFO Depth (INEPnTxFDep) This value is in terms of 32-bit words. Minimum value is 16 Maximum value is 32,768 The power-on reset value of this register is specified as the Largest IN Endpoint FIFO number Depth Programmed values must not exceed the power-on value	RW	0x2000

Bit	Name	Description	Access	Reset
15:0	inepntxfstaddr	IN Endpoint FIFOn Transmit RAM Start Address (INEPnTxFStAddr) This field contains the memory start address For IN endpoint Transmit FIFOn (0<n< = 15). The power-on reset value of this register is specified as the Largest Rx Data FIFO Depth Programmed values must not exceed the power-on value.	RW	0x8000

dieptxf12

Device IN Endpoint Transmit FIFO Size Register 12

Module Instance	Base Address	Register Address
i_usbotg_0_globgrp	0xFFB00000	0xFFB00130
i_usbotg_1_globgrp	0xFFB40000	0xFFB40130

Offset: 0x130

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		inepntxfdep RW 0x2000													
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
inepntxfstaddr RW 0xA000															

dieptxf12 Fields

Bit	Name	Description	Access	Reset
29:16	inepntxfdep	IN Endpoint TxFIFO Depth (INEPnTxFDep) This value is in terms of 32-bit words. Minimum value is 16 Maximum value is 32,768 The power-on reset value of this register is specified as the Largest IN Endpoint FIFO number Depth Programmed values must not exceed the power-on value	RW	0x2000
15:0	inepntxfstaddr	IN Endpoint FIFOn Transmit RAM Start Address (INEPnTxFStAddr) This field contains the memory start address For IN endpoint Transmit FIFOn (0<n< = 15). The power-on reset value of this register is specified as the Largest Rx Data FIFO Depth Programmed values must not exceed the power-on value.	RW	0xA000

dieptxf13

Device IN Endpoint Transmit FIFO Size Register 13

Module Instance	Base Address	Register Address
i_usbotg_0_globgrp	0xFFB00000	0xFFB00134
i_usbotg_1_globgrp	0xFFB40000	0xFFB40134

Offset: 0x134

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		inepntxfdep RW 0x2000													
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
inepntxfstaddr RW 0xC000															

dieptxf13 Fields

Bit	Name	Description	Access	Reset
29:16	inepntxfdep	IN Endpoint TxFIFO Depth (INEPnTxFDep) This value is in terms of 32-bit words. Minimum value is 16 Maximum value is 32,768 The power-on reset value of this register is specified as the Largest IN Endpoint FIFO number Depth Programmed values must not exceed the power-on value	RW	0x2000
15:0	inepntxfstaddr	IN Endpoint FIFOn Transmit RAM Start Address (INEPnTxFStAddr) This field contains the memory start address For IN endpoint Transmit FIFOn (0<n< = 15). The power-on reset value of this register is specified as the Largest Rx Data FIFO Depth Programmed values must not exceed the power-on value.	RW	0xC000

dieptxf14

Device IN Endpoint Transmit FIFO Size Register 14

Module Instance	Base Address	Register Address
i_usbotg_0_globgrp	0xFFB00000	0xFFB00138

Module Instance	Base Address	Register Address
i_usbotg_1_globgrp	0xFFB40000	0xFFB40138

Offset: 0x138

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		inepntxfdep RW 0x2000													
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
inepntxfstaddr RW 0xE000															

dieptxf14 Fields

Bit	Name	Description	Access	Reset
29:16	inepntxfdep	IN Endpoint TxFIFO Depth (INEPnTxFDep) This value is in terms of 32-bit words. Minimum value is 16 Maximum value is 32,768 The power-on reset value of this register is specified as the Largest IN Endpoint FIFO number Depth Programmed values must not exceed the power-on value	RW	0x2000

Bit	Name	Description	Access	Reset
15:0	inepntxfstaddr	IN Endpoint FIFOn Transmit RAM Start Address (INEPnTxFStAddr) This field contains the memory start address For IN endpoint Transmit FIFOn (0<n< = 15). The power-on reset value of this register is specified as the Largest Rx Data FIFO Depth Programmed values must not exceed the power-on value.	RW	0xE000

dieptxf15

Device IN Endpoint Transmit FIFO Size Register 15

Module Instance	Base Address	Register Address
i_usbotg_0_globgrp	0xFFB00000	0xFFB0013C
i_usbotg_1_globgrp	0xFFB40000	0xFFB4013C

Offset: 0x13C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		inepntxfdep RW 0x2000													
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
inepntxfstaddr RW 0x0															

dieptxf15 Fields

Bit	Name	Description	Access	Reset
29:16	inepntxfdep	IN Endpoint TxFIFO Depth (INEPnTxFDep) This value is in terms of 32-bit words. Minimum value is 16 Maximum value is 32,768 The power-on reset value of this register is specified as the Largest IN Endpoint FIFO number Depth Programmed values must not exceed the power-on value	RW	0x2000
15:0	inepntxfstaddr	IN Endpoint FIFOn Transmit RAM Start Address (INEPnTxFStAddr) This field contains the memory start address For IN endpoint Transmit FIFOn (0<n< = 15). The power-on reset value of this register is specified as the Largest Rx Data FIFO Depth Programmed values must not exceed the power-on value.	RW	0x0

usb_hostgrp Address Map

Module Instance	Base Address	End Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB007FF
i_usbotg_1_hostgrp	0xFFB40400	0xFFB407FF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
hcfg on page 18-208	0x0	32	RW	0x200	Host Configuration Register

Register	Offset	Width	Access	Reset Value	Description
hfir on page 18-214	0x4	32	RW	0xEA60	Host Frame Interval Register
hfnum on page 18-216	0x8	32	RO	0x3FFF	Host Frame Number/ Frame Time Remaining Register
hptxsts on page 18-217	0x10	32	RO	0x102000	Host Periodic Transmit FIFO/Queue Status Register
haint on page 18-221	0x14	32	RO	0x0	Host All Channels Interrupt Register
haintmsk on page 18-222	0x18	32	RW	0x0	Host All Channels Interrupt Mask Register
hflbaddr on page 18-223	0x1C	32	RW	0x0	Host Frame List Base Address Register
hprt on page 18-223	0x40	32	RW	0x0	Host Port Control and Status Register
hcchar0 on page 18-231	0x100	32	RW	0x0	Host Channel 0 Characteristics Register
hcsplt0 on page 18-236	0x104	32	RW	0x0	Host Channel 0 Split Control Register
hcint0 on page 18-238	0x108	32	RW	0x0	Host Channel 0 Interrupt Register
hcintmsk0 on page 18-244	0x10C	32	RW	0x0	Host Channel 0 Interrupt Mask Register
hctsiz0 on page 18-248	0x110	32	RW	0x0	Host Channel 0 Transfer Size Register
hcdma0 on page 18-251	0x114	32	RW	0x0	Host Channel 0 DMA Address Register
hcdmab0 on page 18-254	0x11C	32	RW	0x0	Host Channel 0 DMA Buffer Address Register
hcchar1 on page 18-255	0x120	32	RW	0x0	Host Channel 1 Characteristics Register

Register	Offset	Width	Access	Reset Value	Description
hcsplt1 on page 18-260	0x124	32	RW	0x0	Host Channel 1 Split Control Register
hcint1 on page 18-262	0x128	32	RW	0x0	Host Channel 1 Interrupt Register
hcintmsk1 on page 18-268	0x12C	32	RW	0x0	Host Channel 1 Interrupt Mask Register
hctsiz1 on page 18-272	0x130	32	RW	0x0	Host Channel 1 Transfer Size Register
hcdma1 on page 18-275	0x134	32	RW	0x0	Host Channel 1 DMA Address Register
hcdmab1 on page 18-278	0x13C	32	RW	0x0	Host Channel 1 DMA Buffer Address Register
hcchar2 on page 18-279	0x140	32	RW	0x0	Host Channel 2 Characteristics Register
hcsplt2 on page 18-284	0x144	32	RW	0x0	Host Channel 2 Split Control Register
hcint2 on page 18-286	0x148	32	RW	0x0	Host Channel 2 Interrupt Register
hcintmsk2 on page 18-292	0x14C	32	RW	0x0	Host Channel 2 Interrupt Mask Register
hctsiz2 on page 18-296	0x150	32	RW	0x0	Host Channel 2 Transfer Size Register
hcdma2 on page 18-299	0x154	32	RW	0x0	Host Channel 2 DMA Address Register
hcdmab2 on page 18-302	0x15C	32	RW	0x0	Host Channel 2 DMA Buffer Address Register
hcchar3 on page 18-303	0x160	32	RW	0x0	Host Channel 3 Characteristics Register
hcsplt3 on page 18-308	0x164	32	RW	0x0	Host Channel 3 Split Control Register
hcint3 on page 18-310	0x168	32	RW	0x0	Host Channel 3 Interrupt Register

Register	Offset	Width	Access	Reset Value	Description
hcintmsk3 on page 18-316	0x16C	32	RW	0x0	Host Channel 3 Interrupt Mask Register
hctsiz3 on page 18-320	0x170	32	RW	0x0	Host Channel 3 Transfer Size Register
hcdma3 on page 18-323	0x174	32	RW	0x0	Host Channel 3 DMA Address Register
hcdmab3 on page 18-326	0x17C	32	RW	0x0	Host Channel 3 DMA Buffer Address Register
hcchar4 on page 18-327	0x180	32	RW	0x0	Host Channel 4 Characteristics Register
hcsplt4 on page 18-328	0x184	32	RW	0x0	Host Channel 4 Split Control Register
hcint4 on page 18-330	0x188	32	RW	0x0	Host Channel 4 Interrupt Register
hcintmsk4 on page 18-336	0x18C	32	RW	0x0	Host Channel 4 Interrupt Mask Register
hctsiz4 on page 18-340	0x190	32	RW	0x0	Host Channel 4 Transfer Size Register
hcdma4 on page 18-343	0x194	32	RW	0x0	Host Channel 4 DMA Address Register
hcdmab4 on page 18-346	0x19C	32	RW	0x0	Host Channel 4 DMA Buffer Address Register
hcchar5 on page 18-347	0x1A0	32	RW	0x0	Host Channel 5 Characteristics Register
hcsplt5 on page 18-352	0x1A4	32	RW	0x0	Host Channel 5 Split Control Register
hcint5 on page 18-354	0x1A8	32	RW	0x0	Host Channel 5 Interrupt Register
hcintmsk5 on page 18-360	0x1AC	32	RW	0x0	Host Channel 5 Interrupt Mask Register
hctsiz5 on page 18-364	0x1B0	32	RW	0x0	Host Channel 5 Transfer Size Register

Register	Offset	Width	Access	Reset Value	Description
hcdma5 on page 18-367	0x1B4	32	RW	0x0	Host Channel 5 DMA Address Register
hcdmab5 on page 18-370	0x1BC	32	RW	0x0	Host Channel 5 DMA Buffer Address Register
hcchar6 on page 18-371	0x1C0	32	RW	0x0	Host Channel 6 Characteristics Register
hcsplt6 on page 18-376	0x1C4	32	RW	0x0	Host Channel 6 Split Control Register
hcint6 on page 18-378	0x1C8	32	RW	0x0	Host Channel 6 Interrupt Register
hcintmsk6 on page 18-384	0x1CC	32	RW	0x0	Host Channel 6 Interrupt Mask Register
hctsiz6 on page 18-388	0x1D0	32	RW	0x0	Host Channel 6 Transfer Size Register
hcdma6 on page 18-391	0x1D4	32	RW	0x0	Host Channel 6 DMA Address Register
hcdmab6 on page 18-394	0x1DC	32	RW	0x0	Host Channel 6 DMA Buffer Address Register
hcchar7 on page 18-395	0x1E0	32	RW	0x0	Host Channel 7 Characteristics Register
hcsplt7 on page 18-400	0x1E4	32	RW	0x0	Host Channel 7 Split Control Register
hcint7 on page 18-402	0x1E8	32	RW	0x0	Host Channel 7 Interrupt Register
hcintmsk7 on page 18-408	0x1EC	32	RW	0x0	Host Channel 7 Interrupt Mask Register
hctsiz7 on page 18-412	0x1F0	32	RW	0x0	Host Channel 7 Transfer Size Register
hcdma7 on page 18-415	0x1F4	32	RW	0x0	Host Channel 7 DMA Address Register
hcdmab7 on page 18-418	0x1FC	32	RW	0x0	Host Channel 7 DMA Buffer Address Register

Register	Offset	Width	Access	Reset Value	Description
hcchar8 on page 18-419	0x200	32	RW	0x0	Host Channel 8 Characteristics Register
hcsplt8 on page 18-424	0x204	32	RW	0x0	Host Channel 8 Split Control Register
hcint8 on page 18-426	0x208	32	RW	0x0	Host Channel 8 Interrupt Register
hcintmsk8 on page 18-432	0x20C	32	RW	0x0	Host Channel 8 Interrupt Mask Register
hctsiz8 on page 18-436	0x210	32	RW	0x0	Host Channel 8 Transfer Size Register
hcdma8 on page 18-439	0x214	32	RW	0x0	Host Channel 8 DMA Address Register
hcdmab8 on page 18-442	0x21C	32	RW	0x0	Host Channel 8 DMA Buffer Address Register
hcchar9 on page 18-443	0x220	32	RW	0x0	Host Channel 9 Characteristics Register
hcsplt9 on page 18-448	0x224	32	RW	0x0	Host Channel 9 Split Control Register
hcint9 on page 18-450	0x228	32	RW	0x0	Host Channel 9 Interrupt Register
hcintmsk9 on page 18-456	0x22C	32	RW	0x0	Host Channel 9 Interrupt Mask Register
hctsiz9 on page 18-460	0x230	32	RW	0x0	Host Channel 9 Transfer Size Register
hcdma9 on page 18-463	0x234	32	RW	0x0	Host Channel 9 DMA Address Register
hcdmab9 on page 18-466	0x23C	32	RW	0x0	Host Channel 9 DMA Buffer Address Register
hcchar10 on page 18-467	0x240	32	RW	0x0	Host Channel 10 Characteristics Register

Register	Offset	Width	Access	Reset Value	Description
hcsplt10 on page 18-472	0x244	32	RW	0x0	Host Channel 10 Split Control Register
hcint10 on page 18-474	0x248	32	RW	0x0	Host Channel 10 Interrupt Register
hcintmsk10 on page 18-480	0x24C	32	RW	0x0	Host Channel 10 Interrupt Mask Register
hctsiz10 on page 18-484	0x250	32	RW	0x0	Host Channel 10 Transfer Size Register
hcdma10 on page 18-487	0x254	32	RW	0x0	Host Channel 10 DMA Address Register
hcdmab10 on page 18-490	0x25C	32	RW	0x0	Host Channel 10 DMA Buffer Address Register
hcchar11 on page 18-491	0x260	32	RW	0x0	Host Channel 11 Characteristics Register
hcsplt11 on page 18-496	0x264	32	RW	0x0	Host Channel 11 Split Control Register
hcint11 on page 18-498	0x268	32	RW	0x0	Host Channel 11 Interrupt Register
hcintmsk11 on page 18-504	0x26C	32	RW	0x0	Host Channel 11 Interrupt Mask Register
hctsiz11 on page 18-508	0x270	32	RW	0x0	Host Channel 11 Transfer Size Register
hcdma11 on page 18-511	0x274	32	RW	0x0	Host Channel 11 DMA Address Register
hcdmab11 on page 18-514	0x27C	32	RW	0x0	Host Channel 11 DMA Buffer Address Register
hcchar12 on page 18-515	0x280	32	RW	0x0	Host Channel 12 Characteristics Register
hcsplt12 on page 18-520	0x284	32	RW	0x0	Host Channel 12 Split Control Register
hcint12 on page 18-522	0x288	32	RW	0x0	Host Channel 12 Interrupt Register

Register	Offset	Width	Access	Reset Value	Description
hcintmsk12 on page 18-528	0x28C	32	RW	0x0	Host Channel 12 Interrupt Mask Register
hctsiz12 on page 18-532	0x290	32	RW	0x0	Host Channel 12 Transfer Size Register
hcdma12 on page 18-535	0x294	32	RW	0x0	Host Channel 12 DMA Address Register
hcdmab12 on page 18-538	0x29C	32	RW	0x0	Host Channel 12 DMA Buffer Address Register
hcchar13 on page 18-539	0x2A0	32	RW	0x0	Host Channel 13 Characteristics Register
hcsplt13 on page 18-544	0x2A4	32	RW	0x0	Host Channel 13 Split Control Register
hcint13 on page 18-546	0x2A8	32	RW	0x0	Host Channel 13 Interrupt Register
hcintmsk13 on page 18-552	0x2AC	32	RW	0x0	Host Channel 13 Interrupt Mask Register
hctsiz13 on page 18-556	0x2B0	32	RW	0x0	Host Channel 13 Transfer Size Register
hcdma13 on page 18-559	0x2B4	32	RW	0x0	Host Channel 13 DMA Address Register
hcdmab13 on page 18-562	0x2BC	32	RW	0x0	Host Channel 13 DMA Buffer Address Register
hcchar14 on page 18-563	0x2C0	32	RW	0x0	Host Channel 14 Characteristics Register
hcsplt14 on page 18-568	0x2C4	32	RW	0x0	Host Channel 14 Split Control Register
hcint14 on page 18-570	0x2C8	32	RW	0x0	Host Channel 14 Interrupt Register
hcintmsk14 on page 18-576	0x2CC	32	RW	0x0	Host Channel 14 Interrupt Mask Register
hctsiz14 on page 18-580	0x2D0	32	RW	0x0	Host Channel 14 Transfer Size Register

Register	Offset	Width	Access	Reset Value	Description
hcdma14 on page 18-583	0x2D4	32	RW	0x0	Host Channel 14 DMA Address Register
hcdmab14 on page 18-586	0x2DC	32	RW	0x0	Host Channel 14 DMA Buffer Address Register
hcchar15 on page 18-587	0x2E0	32	RW	0x0	Host Channel 15 Characteristics Register
hcsplt15 on page 18-592	0x2E4	32	RW	0x0	Host Channel 15 Split Control Register
hcint15 on page 18-594	0x2E8	32	RW	0x0	Host Channel 15 Interrupt Register
hcintmsk15 on page 18-600	0x2EC	32	RW	0x0	Host Channel 15 Interrupt Mask Register
hctsiz15 on page 18-604	0x2F0	32	RW	0x0	Host Channel 15 Transfer Size Register
hcdma15 on page 18-607	0x2F4	32	RW	0x0	Host Channel 15 DMA Address Register
hcdmab15 on page 18-610	0x2FC	32	RW	0x0	Host Channel 15 DMA Buffer Address Register

usb_hostgrp Summary

Module Instance	Base Address
i_usbotg_0_hostgrp	0xFFB00400
i_usbotg_1_hostgrp	0xFFB40400

Register Address Offset	Bit Fields
i_usbotg_0_hostgrp	

Register Address Offset	Bit Fields															
hcfg 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	modechtimen RW 0x0	Reserved				perschedena RW 0x0	frlisten RW 0x0		descdma RW 0x0	Reserved						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	resvalid RW 0x2								ena32khzs RW 0x0	Reserved				fslssupp RW 0x0	fslspclksel RW 0x0	
hfir 0x4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															hfirrldctrl RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	frint RW 0xEA60															
hfnum 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	frrem RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	frnum RO 0x3FFF															
hptxsts 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	oddevnmframe RO 0x0	chanendpt RO 0x0				type RO 0x0		term RO 0x0	ptxqspcavail RO 0x10							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ptxfspcavail RO 0x2000															
haint 0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	haint RO 0x0															

Register Address Offset	Bit Fields															
haintmsk 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	haintmsk RW 0x0															
hflbaddr 0x1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	hflbaddr RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hflbaddr RW 0x0															
hprt 0x40	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved													prtspd RO 0x0		prttstctl RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	prttstctl RW 0x0			prtpwr RW 0x0	prtlnsts RO 0x0		Reserved	prtrst RW 0x0	prtsusp RW 0x0	prtres RW 0x0	prtovrcurrchng RW 0x0	prtovrcurract RO 0x0	prtenchng RW 0x0	prtena RW 0x0	prtconndet RW 0x0	prtconnsts RO 0x0
hcchar0 0x100	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	chena RW 0x0	chdis RW 0x0	oddfrm RW 0x0	devaddr RW 0x0							ec RW 0x0		eptype RW 0x0		lspddev RW 0x0	Reserved
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	epdir RW 0x0	epnum RW 0x0				mps RW 0x0										
hcsplt0 0x104	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	spltena RW 0x0	Reserved														compsplt RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xactpos RW 0x0		hubaddr RW 0x0							prtaddr RW 0x0						

Register Address Offset	Bit Fields															
hcint0 0x108	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		desc_lst_rollintr RW 0x0	xcs_xact_err RW 0x0	bnaintr RW 0x0	datatglerr RW 0x0	frmovrun RW 0x0	bblerr RW 0x0	xacterr RW 0x0	nyet RW 0x0	ack RW 0x0	nak RW 0x0	stall RW 0x0	ahberr RW 0x0	chhltd RW 0x0	xfercompl RW 0x0
hcintmsk0 0x10C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		frm_lst_rollintrmsk RW 0x0	Reserved	bnaintrmsk RW 0x0	datatglerrmsk RW 0x0	frmovrunmsk RW 0x0	bblerrmsk RW 0x0	xacterrmsk RW 0x0	nyetmsk RW 0x0	ackmsk RW 0x0	nakmsk RW 0x0	stallmsk RW 0x0	ahberrmsk RW 0x0	chhltdmsk RW 0x0	xfercomplmsk RW 0x0
hctsiz0 0x110	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	dopng RW 0x0	pid RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
hcdma0 0x114	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	hcdma0 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hcdma0 RW 0x0															
hcdmab0 0x11C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	hcdmab0 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hcdmab0 RW 0x0															

Register Address Offset	Bit Fields															
hcchar1 0x120	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	chena RW 0x0	chdis RW 0x0	oddfrm RW 0x0	devaddr RW 0x0							ec RW 0x0		eptype RW 0x0		lspddev RW 0x0	Reserved
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	epdir RW 0x0	epnum RW 0x0				mps RW 0x0										
hcsplt1 0x124	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	spltena RW 0x0	Reserved														compsplt RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xactpos RW 0x0		hubaddr RW 0x0							prtaddr RW 0x0						
hcint1 0x128	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		desc_lst_rollintr RW 0x0	xcs_xact_err RW 0x0	bnaintr RW 0x0	datatglerr RW 0x0	frmovrun RW 0x0	bblerr RW 0x0	xacterr RW 0x0	nyet RW 0x0	ack RW 0x0	nak RW 0x0	stall RW 0x0	ahberr RW 0x0	chhltd RW 0x0	xfercompl RW 0x0
hcintmsk1 0x12C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		frm_lst_rollintrmsk RW 0x0	Reserved	bnaintrmsk RW 0x0	datatglerrmsk RW 0x0	frmovrunmsk RW 0x0	bblerrmsk RW 0x0	xacterrmsk RW 0x0	nyetmsk RW 0x0	ackmsk RW 0x0	nakmsk RW 0x0	stallmsk RW 0x0	ahberrmsk RW 0x0	chhltdmsk RW 0x0	xfercomplmsk RW 0x0

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hctsiz1 0x130	dopng RW 0x0	pid RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcdma1 0x134	hcdma1 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hcdma1 RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcdmab1 0x13C	hcdmab1 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hcdmab1 RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcchar2 0x140	chena RW 0x0	chdis RW 0x0	oddfrm RW 0x0	devaddr RW 0x0							ec RW 0x0		eptype RW 0x0		lspddev RW 0x0	Reserved
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	epdir RW 0x0	epnum RW 0x0				mps RW 0x0										
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcsplt2 0x144	spltena RW 0x0	Reserved														compsplt RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xactpos RW 0x0		hubaddr RW 0x0							prtaddr RW 0x0						

Register Address Offset	Bit Fields															
hcint2 0x148	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		desc_lst_rollintr RW 0x0	xcs_xact_err RW 0x0	bnaintr RW 0x0	datatglerr RW 0x0	frmovrun RW 0x0	bblerr RW 0x0	xacterr RW 0x0	nyet RW 0x0	ack RW 0x0	nak RW 0x0	stall RW 0x0	ahberr RW 0x0	chhltd RW 0x0	xfercompl RW 0x0
hcintmsk2 0x14C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		frm_lst_rollintrmsk RW 0x0	Reserved	bnaintrmsk RW 0x0	datatglerrmsk RW 0x0	frmovrunmsk RW 0x0	bblerrmsk RW 0x0	xacterrmsk RW 0x0	nyetmsk RW 0x0	ackmsk RW 0x0	nakmsk RW 0x0	stallmsk RW 0x0	ahberrmsk RW 0x0	chhltdmsk RW 0x0	xfercomplmsk RW 0x0
hctsiz2 0x150	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	dopng RW 0x0	pid RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
hcdma2 0x154	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	hcdma2 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hcdma2 RW 0x0															
hcdmab2 0x15C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	hcdmab2 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hcdmab2 RW 0x0															

Register Address Offset	Bit Fields															
hcchar3 0x160	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	chena RW 0x0	chdis RW 0x0	oddfrm RW 0x0	devaddr RW 0x0							ec RW 0x0		eptype RW 0x0		lspddev RW 0x0	Reserved
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	epdir RW 0x0	epnum RW 0x0				mps RW 0x0										
hcsplt3 0x164	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	spltena RW 0x0	Reserved														compsplt RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xactpos RW 0x0		hubaddr RW 0x0							prtaddr RW 0x0						
hcint3 0x168	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		desc_lst_rollintr RW 0x0	xcs_xact_err RW 0x0	bnaintr RW 0x0	datatglerr RW 0x0	frmovrun RW 0x0	bblerr RW 0x0	xacterr RW 0x0	nyet RW 0x0	ack RW 0x0	nak RW 0x0	stall RW 0x0	ahberr RW 0x0	chhltd RW 0x0	xfercompl RW 0x0
hcintmsk3 0x16C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		frm_lst_rollintrmsk RW 0x0	Reserved	bnaintrmsk RW 0x0	datatglerrmsk RW 0x0	frmovrunmsk RW 0x0	bblerrmsk RW 0x0	xacterrmsk RW 0x0	nyetmsk RW 0x0	ackmsk RW 0x0	nakmsk RW 0x0	stallmsk RW 0x0	ahberrmsk RW 0x0	chhltdmsk RW 0x0	xfercomplmsk RW 0x0

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hctsiz3 0x170	dopng RW 0x0	pid RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
hcdma3 0x174	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	hcdma3 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hcdma3 RW 0x0															
hcdmab3 0x17C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	hcdmab3 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hcdmab3 RW 0x0															
hcchar4 0x180	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	hcdmab4 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hcdmab4 RW 0x0															
hcsplt4 0x184	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	spltena RW 0x0	Reserved														compsplt RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xactpos RW 0x0		hubaddr RW 0x0							prtaddr RW 0x0						

Register Address Offset	Bit Fields															
hcint4 0x188	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		desc_lst_rollintr RW 0x0	xcs_xact_err RW 0x0	bnaintr RW 0x0	datatglerr RW 0x0	frmovrun RW 0x0	bblerr RW 0x0	xacterr RW 0x0	nyet RW 0x0	ack RW 0x0	nak RW 0x0	stall RW 0x0	ahberr RW 0x0	chhltd RW 0x0	xfercompl RW 0x0
hcintmsk4 0x18C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		frm_lst_rollintrmsk RW 0x0	Reserved	bnaintrmsk RW 0x0	datatglerrmsk RW 0x0	frmovrunmsk RW 0x0	bblerrmsk RW 0x0	xacterrmsk RW 0x0	nyetmsk RW 0x0	ackmsk RW 0x0	nakmsk RW 0x0	stallmsk RW 0x0	ahberrmsk RW 0x0	chhltdmsk RW 0x0	xfercomplmsk RW 0x0
hctsiz4 0x190	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	dopng RW 0x0	pid RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
hcdma4 0x194	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	hcdma4 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hcdma4 RW 0x0															
hcdmab4 0x19C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	hcdmab4 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hcdmab4 RW 0x0															

Register Address Offset	Bit Fields															
hcchar5 0x1A0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	chena RW 0x0	chdis RW 0x0	oddfrm RW 0x0	devaddr RW 0x0							ec RW 0x0		eptype RW 0x0		lspddev RW 0x0	Reserved
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	epdir RW 0x0	epnum RW 0x0				mps RW 0x0										
hcsplt5 0x1A4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	spltena RW 0x0	Reserved														compsplt RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xactpos RW 0x0		hubaddr RW 0x0							prtaddr RW 0x0						
hcint5 0x1A8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		desc_lst_rollintr RW 0x0	xcs_xact_err RW 0x0	bnaintr RW 0x0	datatglerr RW 0x0	frmovrun RW 0x0	bblerr RW 0x0	xacterr RW 0x0	nyet RW 0x0	ack RW 0x0	nak RW 0x0	stall RW 0x0	ahberr RW 0x0	chhltd RW 0x0	xfercompl RW 0x0
hcintmsk5 0x1AC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		frm_lst_rollintrmsk RW 0x0	Reserved	bnaintrmsk RW 0x0	datatglerrmsk RW 0x0	frmovrunmsk RW 0x0	bblerrmsk RW 0x0	xacterrmsk RW 0x0	nyetmsk RW 0x0	ackmsk RW 0x0	nakmsk RW 0x0	stallmsk RW 0x0	ahberrmsk RW 0x0	chhltdmsk RW 0x0	xfercomplmsk RW 0x0

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hctsiz5 0x1B0	dopng RW 0x0	pid RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcdma5 0x1B4	hcdma5 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hcdma5 RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcdmab5 0x1BC	hcdmab5 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hcdmab5 RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcchar6 0x1C0	chena RW 0x0	chdis RW 0x0	oddfrm RW 0x0	devaddr RW 0x0							ec RW 0x0		eptype RW 0x0		lspddev RW 0x0	Reserved
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	epdir RW 0x0	epnum RW 0x0				mps RW 0x0										
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcsplt6 0x1C4	spltena RW 0x0	Reserved														compsplt RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xactpos RW 0x0		hubaddr RW 0x0							prtaddr RW 0x0						

Register Address Offset	Bit Fields															
hcint6 0x1C8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		desc_lst_rollintr RW 0x0	xcs_xact_err RW 0x0	bnaintr RW 0x0	datatglerr RW 0x0	frmovrun RW 0x0	bblerr RW 0x0	xacterr RW 0x0	nyet RW 0x0	ack RW 0x0	nak RW 0x0	stall RW 0x0	ahberr RW 0x0	chhltd RW 0x0	xfercompl RW 0x0
hcintmsk6 0x1CC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		frm_lst_rollintrmsk RW 0x0	Reserved	bnaintrmsk RW 0x0	datatglerrmsk RW 0x0	frmovrunmsk RW 0x0	bblerrmsk RW 0x0	xacterrmsk RW 0x0	nyetmsk RW 0x0	ackmsk RW 0x0	nakmsk RW 0x0	stallmsk RW 0x0	ahberrmsk RW 0x0	chhltdmsk RW 0x0	xfercomplmsk RW 0x0
hctsiz6 0x1D0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	dopng RW 0x0	pid RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
hcdma6 0x1D4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	hcdma6 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hcdma6 RW 0x0															
hcdmab6 0x1DC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	hcdmab6 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hcdmab6 RW 0x0															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	chena RW 0x0	chdis RW 0x0	oddfrm RW 0x0	devaddr RW 0x0							ec RW 0x0		eptype RW 0x0		lspddev RW 0x0	Reserved
hcchar7 0x1E0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	epdir RW 0x0	epnum RW 0x0				mps RW 0x0										
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	spltena RW 0x0	Reserved														compsplt RW 0x0
hcsplt7 0x1E4	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xactpos RW 0x0		hubaddr RW 0x0							prtaddr RW 0x0						
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
hcint7 0x1E8	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		desc_lst_rollintr RW 0x0	xcs_xact_err RW 0x0	bnaintr RW 0x0	datatglerr RW 0x0	frmovrun RW 0x0	bblerr RW 0x0	xacterr RW 0x0	nyet RW 0x0	ack RW 0x0	nak RW 0x0	stall RW 0x0	ahberr RW 0x0	chhltd RW 0x0	xfercompl RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
hcintmsk7 0x1EC	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		frm_lst_rollintrmsk RW 0x0	Reserved	bnaintrmsk RW 0x0	datatglerrmsk RW 0x0	frmovrunmsk RW 0x0	bblerrmsk RW 0x0	xacterrmsk RW 0x0	nyetmsk RW 0x0	ackmsk RW 0x0	nakmsk RW 0x0	stallmsk RW 0x0	ahberrmsk RW 0x0	chhltdmsk RW 0x0	xfercomplmsk RW 0x0

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hctsiz7 0x1F0	dopng RW 0x0	pid RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcdma7 0x1F4	hcdma7 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hcdma7 RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcdmab7 0x1FC	hcdmab7 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hcdmab7 RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcchar8 0x200	chena RW 0x0	chdis RW 0x0	oddfrm RW 0x0	devaddr RW 0x0							ec RW 0x0		eptype RW 0x0		lspddev RW 0x0	Reserved
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	epdir RW 0x0	epnum RW 0x0				mps RW 0x0										
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcsplt8 0x204	spltena RW 0x0	Reserved														compsplt RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xactpos RW 0x0		hubaddr RW 0x0							prtaddr RW 0x0						

Register Address Offset	Bit Fields															
hcint8 0x208	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		desc_lst_rollintr RW 0x0	xcs_xact_err RW 0x0	bnaintr RW 0x0	datatglerr RW 0x0	frmovrun RW 0x0	bblerr RW 0x0	xacterr RW 0x0	nyet RW 0x0	ack RW 0x0	nak RW 0x0	stall RW 0x0	ahberr RW 0x0	chhltd RW 0x0	xfercompl RW 0x0
hcintmsk8 0x20C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		frm_lst_rollintrmsk RW 0x0	Reserved	bnaintrmsk RW 0x0	datatglerrmsk RW 0x0	frmovrunmsk RW 0x0	bblerrmsk RW 0x0	xacterrmsk RW 0x0	nyetmsk RW 0x0	ackmsk RW 0x0	nakmsk RW 0x0	stallmsk RW 0x0	ahberrmsk RW 0x0	chhltdmsk RW 0x0	xfercomplmsk RW 0x0
hctsiz8 0x210	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	dopng RW 0x0	pid RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
hcdma8 0x214	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	hcdma8 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hcdma8 RW 0x0															
hcdmab8 0x21C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	hcdmab8 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hcdmab8 RW 0x0															

Register Address Offset	Bit Fields															
hcchar9 0x220	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	chena RW 0x0	chdis RW 0x0	oddfrm RW 0x0	devaddr RW 0x0							ec RW 0x0		eptype RW 0x0		lspddev RW 0x0	Reserved
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	epdir RW 0x0	epnum RW 0x0				mps RW 0x0										
hcsplt9 0x224	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	spltena RW 0x0	Reserved														compsplt RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xactpos RW 0x0		hubaddr RW 0x0							prtaddr RW 0x0						
hcint9 0x228	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		desc_lst_rollintr RW 0x0	xcs_xact_err RW 0x0	bnaintr RW 0x0	datatglerr RW 0x0	frmovrun RW 0x0	bblerr RW 0x0	xacterr RW 0x0	nyet RW 0x0	ack RW 0x0	nak RW 0x0	stall RW 0x0	ahberr RW 0x0	chhltd RW 0x0	xfercompl RW 0x0
hcintmsk9 0x22C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		frm_lst_rollintrmsk RW 0x0	Reserved	bnaintrmsk RW 0x0	datatglerrmsk RW 0x0	frmovrunmsk RW 0x0	bblerrmsk RW 0x0	xacterrmsk RW 0x0	nyetmsk RW 0x0	ackmsk RW 0x0	nakmsk RW 0x0	stallmsk RW 0x0	ahberrmsk RW 0x0	chhltdmsk RW 0x0	xfercomplmsk RW 0x0

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hctsiz9 0x230	dopng RW 0x0	pid RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcdma9 0x234	hcdma9 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hcdma9 RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcdmab9 0x23C	hcdmab9 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hcdmab9 RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcchar10 0x240	chena RW 0x0	chdis RW 0x0	oddfrm RW 0x0	devaddr RW 0x0							ec RW 0x0		eptype RW 0x0		lspddev RW 0x0	Reserved
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	epdir RW 0x0	epnum RW 0x0				mps RW 0x0										
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcsplt10 0x244	spltena RW 0x0	Reserved														compsplt RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xactpos RW 0x0		hubaddr RW 0x0							prtaddr RW 0x0						

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
hcint10 0x248	Reserved		desc_lst_rollintr RW 0x0	xcs_xact_err RW 0x0	bnaintr RW 0x0	datatglerr RW 0x0	frmovrun RW 0x0	bblerr RW 0x0	xacterr RW 0x0	nyet RW 0x0	ack RW 0x0	nak RW 0x0	stall RW 0x0	ahberr RW 0x0	chhltd RW 0x0	xfercompl RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
hcintmsk10 0x24C	Reserved		frm_lst_rollintrmsk RW 0x0	Reserved	bnaintrmsk RW 0x0	datatglerrmsk RW 0x0	frmovrunmsk RW 0x0	bblerrmsk RW 0x0	xacterrmsk RW 0x0	nyetmsk RW 0x0	ackmsk RW 0x0	nakmsk RW 0x0	stallmsk RW 0x0	ahberrmsk RW 0x0	chhltdmsk RW 0x0	xfercomplmsk RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hctsiz10 0x250	dopng RW 0x0	pid RW 0x0	pktcnt RW 0x0											xfersize RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcdma10 0x254	hcdma10 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hcdma10 RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcdmab10 0x25C	hcdmab10 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hcdmab10 RW 0x0															

Register Address Offset	Bit Fields															
hcchar11 0x260	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	chena RW 0x0	chdis RW 0x0	oddfrm RW 0x0	devaddr RW 0x0							ec RW 0x0		eptype RW 0x0		lspddev RW 0x0	Reserved
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	epdir RW 0x0	epnum RW 0x0				mps RW 0x0										
hcsplt11 0x264	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	spltena RW 0x0	Reserved														compsplt RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xactpos RW 0x0		hubaddr RW 0x0							prtaddr RW 0x0						
hcint11 0x268	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		desc_lst_rollintr RW 0x0	xcs_xact_err RW 0x0	bnaintr RW 0x0	datatglerr RW 0x0	frmovrun RW 0x0	bblerr RW 0x0	xacterr RW 0x0	nyet RW 0x0	ack RW 0x0	nak RW 0x0	stall RW 0x0	ahberr RW 0x0	chhltd RW 0x0	xfercompl RW 0x0
hcintmsk11 0x26C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		frm_lst_rollintrmsk RW 0x0	Reserved	bnaintrmsk RW 0x0	datatglerrmsk RW 0x0	frmovrunmsk RW 0x0	bblerrmsk RW 0x0	xacterrmsk RW 0x0	nyetmsk RW 0x0	ackmsk RW 0x0	nakmsk RW 0x0	stallmsk RW 0x0	ahberrmsk RW 0x0	chhltdmsk RW 0x0	xfercomplmsk RW 0x0

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hctsiz11 0x270	dopng RW 0x0	pid RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcdma11 0x274	hcdma11 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hcdma11 RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcdmab11 0x27C	hcdmab11 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hcdmab11 RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcchar12 0x280	chena RW 0x0	chdis RW 0x0	oddfrm RW 0x0	devaddr RW 0x0							ec RW 0x0		eptype RW 0x0		lspddev RW 0x0	Reserved
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	epdir RW 0x0	epnum RW 0x0				mps RW 0x0										
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcsplt12 0x284	spltena RW 0x0	Reserved														compsplt RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xactpos RW 0x0		hubaddr RW 0x0							prtaddr RW 0x0						

Register Address Offset	Bit Fields															
hcint12 0x288	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		desc_lst_rollintr RW 0x0	xcs_xact_err RW 0x0	bnaintr RW 0x0	datatglerr RW 0x0	frmovrun RW 0x0	bblerr RW 0x0	xacterr RW 0x0	nyet RW 0x0	ack RW 0x0	nak RW 0x0	stall RW 0x0	ahberr RW 0x0	chhltd RW 0x0	xfercompl RW 0x0
hcintmsk12 0x28C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		frm_lst_rollintrmsk RW 0x0	Reserved	bnaintrmsk RW 0x0	datatglerrmsk RW 0x0	frmovrunmsk RW 0x0	bblerrmsk RW 0x0	xacterrmsk RW 0x0	nyetmsk RW 0x0	ackmsk RW 0x0	nakmsk RW 0x0	stallmsk RW 0x0	ahberrmsk RW 0x0	chhltdmsk RW 0x0	xfercomplmsk RW 0x0
hctsiz12 0x290	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	dopng RW 0x0	pid RW 0x0			pktcnt RW 0x0									xfersize RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
hcdma12 0x294	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	hcdma12 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hcdma12 RW 0x0															
hcdmab12 0x29C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	hcdmab12 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hcdmab12 RW 0x0															

Register Address Offset	Bit Fields															
hcchar13 0x2A0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	chena RW 0x0	chdis RW 0x0	oddfrm RW 0x0	devaddr RW 0x0							ec RW 0x0		eptype RW 0x0		lspddev RW 0x0	Reserved
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	epdir RW 0x0	epnum RW 0x0				mps RW 0x0										
hcsplt13 0x2A4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	spltena RW 0x0	Reserved														compsplt RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xactpos RW 0x0		hubaddr RW 0x0							prtaddr RW 0x0						
hcint13 0x2A8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		desc_lst_rollintr RW 0x0	xcs_xact_err RW 0x0	bnaintr RW 0x0	datatglerr RW 0x0	frmovrun RW 0x0	bblerr RW 0x0	xacterr RW 0x0	nyet RW 0x0	ack RW 0x0	nak RW 0x0	stall RW 0x0	ahberr RW 0x0	chhltd RW 0x0	xfercompl RW 0x0
hcintmsk13 0x2AC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		frm_lst_rollintrmsk RW 0x0	Reserved	bnaintrmsk RW 0x0	datatglerrmsk RW 0x0	frmovrunmsk RW 0x0	bblerrmsk RW 0x0	xacterrmsk RW 0x0	nyetmsk RW 0x0	ackmsk RW 0x0	nakmsk RW 0x0	stallmsk RW 0x0	ahberrmsk RW 0x0	chhltdmsk RW 0x0	xfercomplmsk RW 0x0

Register Address Offset	Bit Fields															
hctsiz13 0x2B0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	dopng RW 0x0	pid RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
hcdma13 0x2B4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	hcdma13 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hcdma13 RW 0x0															
hcdmab13 0x2BC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	hcdmab13 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hcdmab13 RW 0x0															
hcchar14 0x2C0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	chena RW 0x0	chdis RW 0x0	oddfrm RW 0x0	devaddr RW 0x0							ec RW 0x0		eptype RW 0x0		lspddev RW 0x0	Reserved
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	epdir RW 0x0	epnum RW 0x0				mps RW 0x0										
hcsplt14 0x2C4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	spltena RW 0x0	Reserved														compsplt RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xactpos RW 0x0		hubaddr RW 0x0							prtaddr RW 0x0						

Register Address Offset	Bit Fields															
hcint14 0x2C8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		desc_lst_rollintr RW 0x0	xcs_xact_err RW 0x0	bnaintr RW 0x0	datatglerr RW 0x0	frmovrun RW 0x0	bblerr RW 0x0	xacterr RW 0x0	nyet RW 0x0	ack RW 0x0	nak RW 0x0	stall RW 0x0	ahberr RW 0x0	chhltd RW 0x0	xfercompl RW 0x0
hcintmsk14 0x2CC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		frm_lst_rollintrmsk RW 0x0	Reserved	bnaintrmsk RW 0x0	datatglerrmsk RW 0x0	frmovrunmsk RW 0x0	bblerrmsk RW 0x0	xacterrmsk RW 0x0	nyetmsk RW 0x0	ackmsk RW 0x0	nakmsk RW 0x0	stallmsk RW 0x0	ahberrmsk RW 0x0	chhltdmsk RW 0x0	xfercomplmsk RW 0x0
hctsiz14 0x2D0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	dopng RW 0x0	pid RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
hcdma14 0x2D4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	hcdma14 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hcdma14 RW 0x0															
hcdmab14 0x2DC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	hcdmab14 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hcdmab14 RW 0x0															

Register Address Offset	Bit Fields															
hcchar15 0x2E0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	chena RW 0x0	chdis RW 0x0	oddfrm RW 0x0	devaddr RW 0x0							ec RW 0x0		eptype RW 0x0		lspddev RW 0x0	Reserved
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	epdir RW 0x0	epnum RW 0x0				mps RW 0x0										
hcsplt15 0x2E4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	spltena RW 0x0	Reserved														compsplt RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xactpos RW 0x0		hubaddr RW 0x0							prtaddr RW 0x0						
hcint15 0x2E8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		desc_lst_rollintr RW 0x0	xcs_xact_err RW 0x0	bnaintr RW 0x0	datatglerr RW 0x0	frmovrun RW 0x0	bblerr RW 0x0	xacterr RW 0x0	nyet RW 0x0	ack RW 0x0	nak RW 0x0	stall RW 0x0	ahberr RW 0x0	chhltd RW 0x0	xfercompl RW 0x0
hcintmsk15 0x2EC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		frm_lst_rollintrmsk RW 0x0	Reserved	bnaintrmsk RW 0x0	datatglerrmsk RW 0x0	frmovrunmsk RW 0x0	bblerrmsk RW 0x0	xacterrmsk RW 0x0	nyetmsk RW 0x0	ackmsk RW 0x0	nakmsk RW 0x0	stallmsk RW 0x0	ahberrmsk RW 0x0	chhltdmsk RW 0x0	xfercomplmsk RW 0x0

Register Address Offset	Bit Fields															
hctsiz15 0x2F0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	dopng RW 0x0	pid RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
hcdma15 0x2F4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	hcdma15 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hcdma15 RW 0x0															
hcdmab15 0x2FC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	hcdmab15 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hcdmab15 RW 0x0															
i_usbotg_1_hostgrp																
hcfg 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	modechtimen RW 0x0	Reserved				perschedena RW 0x0	frlisten RW 0x0		descdma RW 0x0	Reserved						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	resvalid RW 0x2								ena32khzs RW 0x0	Reserved				fslssupp RW 0x0	fslspclksel RW 0x0	
hfir 0x4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															hfirrldctrl RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	frint RW 0xEA60															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hfnum 0x8	frrem RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	frnum RO 0x3FFF															
hptxsts 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	oddevnmframe RO 0x0	chanendpt RO 0x0				type RO 0x0		term RO 0x0	ptxqspcavail RO 0x10							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ptxfspcavail RO 0x2000															
haint 0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	haint RO 0x0															
haintmsk 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	haintmsk RW 0x0															
hflbaddr 0x1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	hflbaddr RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hflbaddr RW 0x0															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved													prtspd RO 0x0		prttstctl RW 0x0
hprt 0x40	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	prttstctl RW 0x0			prtpwr RW 0x0	prtlnsts RO 0x0		Reserved	prtrst RW 0x0	prtsusp RW 0x0	prtres RW 0x0	prtovrcurrchng RW 0x0	prtovrcurract RO 0x0	prtenchng RW 0x0	prtena RW 0x0	prtconndet RW 0x0	prtconnsts RO 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	chena RW 0x0	chdis RW 0x0	oddfrm RW 0x0	devaddr RW 0x0							ec RW 0x0		eptype RW 0x0		lspddev RW 0x0	Reserved
hcchar0 0x100	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	epdir RW 0x0	epnum RW 0x0				mps RW 0x0										
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	spltena RW 0x0	Reserved														compsplt RW 0x0
hcsplt0 0x104	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xactpos RW 0x0		hubaddr RW 0x0							prtaddr RW 0x0						
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
hcint0 0x108	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		desc_lst_rollintr RW 0x0	xcs_xact_err RW 0x0	bnaintr RW 0x0	datatglerr RW 0x0	frmovrun RW 0x0	bblerr RW 0x0	xacterr RW 0x0	nyet RW 0x0	ack RW 0x0	nak RW 0x0	stall RW 0x0	ahberr RW 0x0	chhltd RW 0x0	xfercompl RW 0x0

Register Address Offset	Bit Fields															
hcintmsk0 0x10C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		frm_lst_rollintrmsk RW 0x0	Reserved	bnaintrmsk RW 0x0	datatglerrmsk RW 0x0	frmovrunmsk RW 0x0	bblerrmsk RW 0x0	xacterrmsk RW 0x0	nyetmsk RW 0x0	ackmsk RW 0x0	nakmsk RW 0x0	stallmsk RW 0x0	ahberrmsk RW 0x0	chhltdmsk RW 0x0	xfercomplmsk RW 0x0
hctsiz0 0x110	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	dopng RW 0x0	pid RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
hcdma0 0x114	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	hcdma0 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hcdma0 RW 0x0															
hcdmab0 0x11C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	hcdmab0 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hcdmab0 RW 0x0															
hcchar1 0x120	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	chena RW 0x0	chdis RW 0x0	oddfrm RW 0x0	devaddr RW 0x0							ec RW 0x0		eptype RW 0x0		lspddev RW 0x0	Reserved
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	epdir RW 0x0	epnum RW 0x0				mps RW 0x0										

Register Address Offset	Bit Fields															
hcsplt1 0x124	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	spltena RW 0x0	Reserved														compsplt RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xactpos RW 0x0		hubaddr RW 0x0							prtaddr RW 0x0						
hcint1 0x128	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		desc_lst_rollintr RW 0x0	xcs_xact_err RW 0x0	bnaintr RW 0x0	datatglerr RW 0x0	frmovrun RW 0x0	bblerr RW 0x0	xacterr RW 0x0	nyet RW 0x0	ack RW 0x0	nak RW 0x0	stall RW 0x0	ahberr RW 0x0	chhltd RW 0x0	xfercompl RW 0x0
hcintmsk1 0x12C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		frm_lst_rollintrmsk RW 0x0	Reserved	bnaintrmsk RW 0x0	datatglerrmsk RW 0x0	frmovrunmsk RW 0x0	bblerrmsk RW 0x0	xacterrmsk RW 0x0	nyetmsk RW 0x0	ackmsk RW 0x0	nakmsk RW 0x0	stallmsk RW 0x0	ahberrmsk RW 0x0	chhltdmsk RW 0x0	xfercomplmsk RW 0x0
hctsiz1 0x130	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	dopng RW 0x0	pid RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
hcdma1 0x134	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	hcdma1 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hcdma1 RW 0x0															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcdmab1	hcdmab1 RW 0x0															
0x13C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hcdmab1 RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcchar2	chena RW 0x0	chdis RW 0x0	oddfrm RW 0x0	devaddr RW 0x0							ec RW 0x0		eptype RW 0x0		lspddev RW 0x0	Reserved
0x140	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	epdir RW 0x0	epnum RW 0x0				mps RW 0x0										
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcsplt2	spltena RW 0x0	Reserved														compsplt RW 0x0
0x144	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xactpos RW 0x0		hubaddr RW 0x0							prtaddr RW 0x0						
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcint2	Reserved															
0x148	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		desc_lst_rollintr RW 0x0	xcs_xact_err RW 0x0	bnaintr RW 0x0	datatglerr RW 0x0	frmovrun RW 0x0	bblerr RW 0x0	xacterr RW 0x0	nyet RW 0x0	ack RW 0x0	nak RW 0x0	stall RW 0x0	ahberr RW 0x0	chhltd RW 0x0	xfercompl RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcintmsk2	Reserved															
0x14C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		frm_lst_rollintrmsk RW 0x0	Reserved	bnaintrmsk RW 0x0	datatglerrmsk RW 0x0	frmovrunmsk RW 0x0	bblerrmsk RW 0x0	xacterrmsk RW 0x0	nyetmsk RW 0x0	ackmsk RW 0x0	nakmsk RW 0x0	stallmsk RW 0x0	ahberrmsk RW 0x0	chhltdmsk RW 0x0	xfercomplmsk RW 0x0

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hctsiz2 0x150	dopng RW 0x0	pid RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcdma2 0x154	hcdma2 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hcdma2 RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcdmab2 0x15C	hcdmab2 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hcdmab2 RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcchar3 0x160	chena RW 0x0	chdis RW 0x0	oddfrm RW 0x0	devaddr RW 0x0							ec RW 0x0		eptype RW 0x0		lspddev RW 0x0	Reserved
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	epdir RW 0x0	epnum RW 0x0				mps RW 0x0										
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcsplt3 0x164	spltena RW 0x0	Reserved														compsplt RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xactpos RW 0x0		hubaddr RW 0x0							prtaddr RW 0x0						

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
hcint3 0x168	Reserved		desc_lst_rollintr RW 0x0	xcs_xact_err RW 0x0	bnaintr RW 0x0	datatglerr RW 0x0	frmovrun RW 0x0	bblerr RW 0x0	xacterr RW 0x0	nyet RW 0x0	ack RW 0x0	nak RW 0x0	stall RW 0x0	ahberr RW 0x0	chhltd RW 0x0	xfercompl RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
hcintmsk3 0x16C	Reserved		frm_lst_rollintrmsk RW 0x0	Reserved	bnaintrmsk RW 0x0	datatglerrmsk RW 0x0	frmovrunmsk RW 0x0	bblerrmsk RW 0x0	xacterrmsk RW 0x0	nyetmsk RW 0x0	ackmsk RW 0x0	nakmsk RW 0x0	stallmsk RW 0x0	ahberrmsk RW 0x0	chhltdmsk RW 0x0	xfercomplmsk RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hctsiz3 0x170	dopng RW 0x0	pid RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcdma3 0x174	hcdma3 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hcdma3 RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcdmab3 0x17C	hcdmab3 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hcdmab3 RW 0x0															

Register Address Offset	Bit Fields															
hcchar4 0x180	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	hcdmab4 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hcdmab4 RW 0x0															
hcsplt4 0x184	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	spltena RW 0x0	Reserved														compsplt RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xactpos RW 0x0		hubaddr RW 0x0							prtaddr RW 0x0						
hcint4 0x188	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		desc_lst_rollintr RW 0x0	xcs_xact_err RW 0x0	bnaintr RW 0x0	datatglerr RW 0x0	frmovrun RW 0x0	bblerr RW 0x0	xacterr RW 0x0	nyet RW 0x0	ack RW 0x0	nak RW 0x0	stall RW 0x0	ahberr RW 0x0	chhltd RW 0x0	xfercompl RW 0x0
hcintmsk4 0x18C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		frm_lst_rollintrmsk RW 0x0	Reserved	bnaintrmsk RW 0x0	datatglerrmsk RW 0x0	frmovrunmsk RW 0x0	bblerrmsk RW 0x0	xacterrmsk RW 0x0	nyetmsk RW 0x0	ackmsk RW 0x0	nakmsk RW 0x0	stallmsk RW 0x0	ahberrmsk RW 0x0	chhltdmsk RW 0x0	xfercomplmsk RW 0x0
hctsiz4 0x190	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	dopng RW 0x0	pid RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcdma4 0x194	hcdma4 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hcdma4 RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcdmab4 0x19C	hcdmab4 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hcdmab4 RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcchar5 0x1A0	chena RW 0x0	chdis RW 0x0	oddfrm RW 0x0	devaddr RW 0x0							ec RW 0x0		eptype RW 0x0		lspddev RW 0x0	Reserved
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	epdir RW 0x0	epnum RW 0x0				mps RW 0x0										
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcsplt5 0x1A4	spltena RW 0x0	Reserved														compsplt RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xactpos RW 0x0	hubaddr RW 0x0								prtaddr RW 0x0						
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcint5 0x1A8	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		desc_lst_rollintr RW 0x0	xcs_xact_err RW 0x0	bnaintr RW 0x0	datatglerr RW 0x0	frmovrun RW 0x0	bblerr RW 0x0	xacterr RW 0x0	nyet RW 0x0	ack RW 0x0	nak RW 0x0	stall RW 0x0	ahberr RW 0x0	chhltd RW 0x0	xfercompl RW 0x0

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
hcintmsk5 0x1AC	Reserved		frm_lst_rollintrmsk RW 0x0	Reserved	bnaintrmsk RW 0x0	datatglerrmsk RW 0x0	frmovrunmsk RW 0x0	bblerrmsk RW 0x0	xacterrmsk RW 0x0	nyetmsk RW 0x0	ackmsk RW 0x0	nakmsk RW 0x0	stallmsk RW 0x0	ahberrmsk RW 0x0	chhltdmsk RW 0x0	xfercomplmsk RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hctsiz5 0x1B0	dopng RW 0x0	pid RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcdma5 0x1B4	hcdma5 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hcdma5 RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcdmab5 0x1BC	hcdmab5 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hcdmab5 RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcchar6 0x1C0	chena RW 0x0	chdis RW 0x0	oddfrm RW 0x0	devaddr RW 0x0							ec RW 0x0		eptype RW 0x0		lspddev RW 0x0	Reserved
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	epdir RW 0x0	epnum RW 0x0				mps RW 0x0										

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcsplt6 0x1C4	splt ena RW 0x0	Reserved														compsplt RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xactpos RW 0x0		hubaddr RW 0x0							prtaddr RW 0x0						
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcint6 0x1C8	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		desc_lst_rollintr RW 0x0	xcs_xact_err RW 0x0	bnaintr RW 0x0	datatglerr RW 0x0	frmovrun RW 0x0	bblerr RW 0x0	xacterr RW 0x0	nyet RW 0x0	ack RW 0x0	nak RW 0x0	stall RW 0x0	ahberr RW 0x0	chhltd RW 0x0	xfercompl RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcintmsk6 0x1CC	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		frm_lst_rollintrmsk RW 0x0	Reserved	bnaintrmsk RW 0x0	datatglerrmsk RW 0x0	frmovrunmsk RW 0x0	bblerrmsk RW 0x0	xacterrmsk RW 0x0	nyetmsk RW 0x0	ackmsk RW 0x0	nakmsk RW 0x0	stallmsk RW 0x0	ahberrmsk RW 0x0	chhltdmsk RW 0x0	xfercomplmsk RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hctsiz6 0x1D0	dopng RW 0x0	pid RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcdma6 0x1D4	hcdma6 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hcdma6 RW 0x0															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcdmab6 0x1DC	hcdmab6 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hcdmab6 RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcchar7 0x1E0	chena RW 0x0	chdis RW 0x0	oddfrm RW 0x0	devaddr RW 0x0							ec RW 0x0		eptype RW 0x0		lspddev RW 0x0	Reserved
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	epdir RW 0x0	epnum RW 0x0				mps RW 0x0										
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcsplt7 0x1E4	spltena RW 0x0	Reserved														compsplt RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xactpos RW 0x0		hubaddr RW 0x0							prtaddr RW 0x0						
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcint7 0x1E8	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		desc_lst_rollintr RW 0x0	xcs_xact_err RW 0x0	bnaintr RW 0x0	datatglerr RW 0x0	frmovrun RW 0x0	bblerr RW 0x0	xacterr RW 0x0	nyet RW 0x0	ack RW 0x0	nak RW 0x0	stall RW 0x0	ahberr RW 0x0	chhltd RW 0x0	xfercompl RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcintmsk7 0x1EC	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		frm_lst_rollintrmsk RW 0x0	Reserved	bnaintrmsk RW 0x0	datatglerrmsk RW 0x0	frmovrunmsk RW 0x0	bblerrmsk RW 0x0	xacterrmsk RW 0x0	nyetmsk RW 0x0	ackmsk RW 0x0	nakmsk RW 0x0	stallmsk RW 0x0	ahberrmsk RW 0x0	chhltdmsk RW 0x0	xfercomplmsk RW 0x0

Register Address Offset	Bit Fields															
hctsiz7 0x1F0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	dopng RW 0x0	pid RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
hcdma7 0x1F4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	hcdma7 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hcdma7 RW 0x0															
hcdmab7 0x1FC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	hcdmab7 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hcdmab7 RW 0x0															
hcchar8 0x200	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	chena RW 0x0	chdis RW 0x0	oddfrm RW 0x0	devaddr RW 0x0							ec RW 0x0		eptype RW 0x0		lspddev RW 0x0	Reserved
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	epdir RW 0x0	epnum RW 0x0				mps RW 0x0										
hcsplt8 0x204	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	spltena RW 0x0	Reserved														compsplt RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xactpos RW 0x0		hubaddr RW 0x0							prtaddr RW 0x0						

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
hcint8 0x208	Reserved		desc_lst_rollintr RW 0x0	xcs_xact_err RW 0x0	bnaintr RW 0x0	datatglerr RW 0x0	frmovrun RW 0x0	bblerr RW 0x0	xacterr RW 0x0	nyet RW 0x0	ack RW 0x0	nak RW 0x0	stall RW 0x0	ahberr RW 0x0	chhltd RW 0x0	xfercompl RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
hcintmsk8 0x20C	Reserved		frm_lst_rollintrmsk RW 0x0	Reserved	bnaintrmsk RW 0x0	datatglerrmsk RW 0x0	frmovrunmsk RW 0x0	bblerrmsk RW 0x0	xacterrmsk RW 0x0	nyetmsk RW 0x0	ackmsk RW 0x0	nakmsk RW 0x0	stallmsk RW 0x0	ahberrmsk RW 0x0	chhltdmsk RW 0x0	xfercomplmsk RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hctsiz8 0x210	dopng RW 0x0	pid RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcdma8 0x214	hcdma8 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hcdma8 RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcdmab8 0x21C	hcdmab8 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hcdmab8 RW 0x0															

Register Address Offset	Bit Fields															
hcchar9 0x220	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	chena RW 0x0	chdis RW 0x0	oddfrm RW 0x0	devaddr RW 0x0							ec RW 0x0		eptype RW 0x0		lspddev RW 0x0	Reserved
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	epdir RW 0x0	epnum RW 0x0				mps RW 0x0										
hcsplt9 0x224	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	spltena RW 0x0	Reserved														compsplt RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xactpos RW 0x0		hubaddr RW 0x0							prtaddr RW 0x0						
hcint9 0x228	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		desc_lst_rollintr RW 0x0	xcs_xact_err RW 0x0	bnaintr RW 0x0	datatglerr RW 0x0	frmovrun RW 0x0	bblerr RW 0x0	xacterr RW 0x0	nyet RW 0x0	ack RW 0x0	nak RW 0x0	stall RW 0x0	ahberr RW 0x0	chhltd RW 0x0	xfercompl RW 0x0
hcintmsk9 0x22C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		frm_lst_rollintrmsk RW 0x0	Reserved	bnaintrmsk RW 0x0	datatglerrmsk RW 0x0	frmovrunmsk RW 0x0	bblerrmsk RW 0x0	xacterrmsk RW 0x0	nyetmsk RW 0x0	ackmsk RW 0x0	nakmsk RW 0x0	stallmsk RW 0x0	ahberrmsk RW 0x0	chhltdmsk RW 0x0	xfercomplmsk RW 0x0

Register Address Offset	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hctsiz9 0x230	dopng RW 0x0	pid RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
hcdma9 0x234	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	hcdma9 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hcdma9 RW 0x0															
hcdmab9 0x23C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	hcdmab9 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hcdmab9 RW 0x0															
hcchar10 0x240	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	chena RW 0x0	chdis RW 0x0	oddfrm RW 0x0	devaddr RW 0x0							ec RW 0x0		eptype RW 0x0		lspddev RW 0x0	Reserved
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	epdir RW 0x0	epnum RW 0x0				mps RW 0x0										
hcsplt10 0x244	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	spltena RW 0x0	Reserved														compsplt RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xactpos RW 0x0		hubaddr RW 0x0							prtaddr RW 0x0						

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
hcint10 0x248	Reserved		desc_lst_rollintr RW 0x0	xcs_xact_err RW 0x0	bnaintr RW 0x0	datatglerr RW 0x0	frmovrun RW 0x0	bblerr RW 0x0	xacterr RW 0x0	nyet RW 0x0	ack RW 0x0	nak RW 0x0	stall RW 0x0	ahberr RW 0x0	chhltd RW 0x0	xfercompl RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
hcintmsk10 0x24C	Reserved		frm_lst_rollintrmsk RW 0x0	Reserved	bnaintrmsk RW 0x0	datatglerrmsk RW 0x0	frmovrunmsk RW 0x0	bblerrmsk RW 0x0	xacterrmsk RW 0x0	nyetmsk RW 0x0	ackmsk RW 0x0	nakmsk RW 0x0	stallmsk RW 0x0	ahberrmsk RW 0x0	chhltdmsk RW 0x0	xfercomplmsk RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hctsiz10 0x250	dopng RW 0x0	pid RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcdma10 0x254	hcdma10 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hcdma10 RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcdmab10 0x25C	hcdmab10 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hcdmab10 RW 0x0															

Register Address Offset	Bit Fields															
hcchar11 0x260	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	chena RW 0x0	chdis RW 0x0	oddfrm RW 0x0	devaddr RW 0x0							ec RW 0x0		eptype RW 0x0		lspddev RW 0x0	Reserved
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	epdir RW 0x0	epnum RW 0x0				mps RW 0x0										
hcsplt11 0x264	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	spltena RW 0x0	Reserved														compsplt RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xactpos RW 0x0		hubaddr RW 0x0							prtaddr RW 0x0						
hcint11 0x268	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		desc_lst_rollintr RW 0x0	xcs_xact_err RW 0x0	bnaintr RW 0x0	datatglerr RW 0x0	frmovrun RW 0x0	bblerr RW 0x0	xacterr RW 0x0	nyet RW 0x0	ack RW 0x0	nak RW 0x0	stall RW 0x0	ahberr RW 0x0	chhltd RW 0x0	xfercompl RW 0x0
hcintmsk11 0x26C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		frm_lst_rollintrmsk RW 0x0	Reserved	bnaintrmsk RW 0x0	datatglerrmsk RW 0x0	frmovrunmsk RW 0x0	bblerrmsk RW 0x0	xacterrmsk RW 0x0	nyetmsk RW 0x0	ackmsk RW 0x0	nakmsk RW 0x0	stallmsk RW 0x0	ahberrmsk RW 0x0	chhltdmsk RW 0x0	xfercomplmsk RW 0x0

Register Address Offset	Bit Fields															
hctsiz11 0x270	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	dopng RW 0x0	pid RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
hcdma11 0x274	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	hcdma11 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hcdma11 RW 0x0															
hcdmab11 0x27C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	hcdmab11 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hcdmab11 RW 0x0															
hcchar12 0x280	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	chena RW 0x0	chdis RW 0x0	oddfrm RW 0x0	devaddr RW 0x0							ec RW 0x0		eptype RW 0x0		lspddev RW 0x0	Reserved
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	epdir RW 0x0	epnum RW 0x0				mps RW 0x0										
hcsplt12 0x284	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	spltena RW 0x0	Reserved														compsplt RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xactpos RW 0x0		hubaddr RW 0x0							prtaddr RW 0x0						

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
hcint12 0x288	Reserved		desc_lst_rollintr RW 0x0	xcs_xact_err RW 0x0	bnaintr RW 0x0	datatglerr RW 0x0	frmovrun RW 0x0	bblerr RW 0x0	xacterr RW 0x0	nyet RW 0x0	ack RW 0x0	nak RW 0x0	stall RW 0x0	ahberr RW 0x0	chhltd RW 0x0	xfercompl RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
hcintmsk12 0x28C	Reserved		frm_lst_rollintrmsk RW 0x0	Reserved	bnaintrmsk RW 0x0	datatglerrmsk RW 0x0	frmovrunmsk RW 0x0	bblerrmsk RW 0x0	xacterrmsk RW 0x0	nyetmsk RW 0x0	ackmsk RW 0x0	nakmsk RW 0x0	stallmsk RW 0x0	ahberrmsk RW 0x0	chhltdmsk RW 0x0	xfercomplmsk RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hctsiz12 0x290	dopng RW 0x0	pid RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcdma12 0x294	hcdma12 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hcdma12 RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcdmab12 0x29C	hcdmab12 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hcdmab12 RW 0x0															

Register Address Offset	Bit Fields															
hcchar13 0x2A0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	chena RW 0x0	chdis RW 0x0	oddfrm RW 0x0	devaddr RW 0x0							ec RW 0x0		eptype RW 0x0		lspddev RW 0x0	Reserved
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	epdir RW 0x0	epnum RW 0x0				mps RW 0x0										
hcsplt13 0x2A4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	spltena RW 0x0	Reserved														compsplt RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xactpos RW 0x0		hubaddr RW 0x0							prtaddr RW 0x0						
hcint13 0x2A8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		desc_lst_rollintr RW 0x0	xcs_xact_err RW 0x0	bnaintr RW 0x0	datatglerr RW 0x0	frmovrun RW 0x0	bblerr RW 0x0	xacterr RW 0x0	nyet RW 0x0	ack RW 0x0	nak RW 0x0	stall RW 0x0	ahberr RW 0x0	chhltd RW 0x0	xfercompl RW 0x0
hcintmsk13 0x2AC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		frm_lst_rollintrmsk RW 0x0	Reserved	bnaintrmsk RW 0x0	datatglerrmsk RW 0x0	frmovrunmsk RW 0x0	bblerrmsk RW 0x0	xacterrmsk RW 0x0	nyetmsk RW 0x0	ackmsk RW 0x0	nakmsk RW 0x0	stallmsk RW 0x0	ahberrmsk RW 0x0	chhltdmsk RW 0x0	xfercomplmsk RW 0x0

Register Address Offset	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hctsiz13 0x2B0	dopng RW 0x0	pid RW 0x0			pktcnt RW 0x0									xfersize RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
hcdma13 0x2B4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	hcdma13 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hcdma13 RW 0x0															
hcdmab13 0x2BC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	hcdmab13 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hcdmab13 RW 0x0															
hcchar14 0x2C0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	chena RW 0x0	chdis RW 0x0	oddfrm RW 0x0	devaddr RW 0x0							ec RW 0x0		eptype RW 0x0		lspddev RW 0x0	Reserved
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	epdir RW 0x0	epnum RW 0x0				mps RW 0x0										
hcsplt14 0x2C4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	spltena RW 0x0	Reserved														compsplt RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xactpos RW 0x0		hubaddr RW 0x0							prtaddr RW 0x0						

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
hcint14 0x2C8	Reserved		desc_lst_rollintr RW 0x0	xcs_xact_err RW 0x0	bnaintr RW 0x0	datatglerr RW 0x0	frmovrun RW 0x0	bblerr RW 0x0	xacterr RW 0x0	nyet RW 0x0	ack RW 0x0	nak RW 0x0	stall RW 0x0	ahberr RW 0x0	chhltd RW 0x0	xfercompl RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
hcintmsk14 0x2CC	Reserved		frm_lst_rollintrmsk RW 0x0	Reserved	bnaintrmsk RW 0x0	datatglerrmsk RW 0x0	frmovrunmsk RW 0x0	bblerrmsk RW 0x0	xacterrmsk RW 0x0	nyetmsk RW 0x0	ackmsk RW 0x0	nakmsk RW 0x0	stallmsk RW 0x0	ahberrmsk RW 0x0	chhltdmsk RW 0x0	xfercomplmsk RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hctsiz14 0x2D0	dopng RW 0x0	pid RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcdma14 0x2D4	hcdma14 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hcdma14 RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcdmab14 0x2DC	hcdmab14 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hcdmab14 RW 0x0															

Register Address Offset	Bit Fields															
hcchar15 0x2E0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	chena RW 0x0	chdis RW 0x0	oddfrm RW 0x0	devaddr RW 0x0							ec RW 0x0		eptype RW 0x0		lspddev RW 0x0	Reserved
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	epdir RW 0x0	epnum RW 0x0				mps RW 0x0										
hcsplt15 0x2E4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	spltena RW 0x0	Reserved														compsplt RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xactpos RW 0x0		hubaddr RW 0x0							prtaddr RW 0x0						
hcint15 0x2E8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		desc_lst_rollintr RW 0x0	xcs_xact_err RW 0x0	bnaintr RW 0x0	datatglerr RW 0x0	frmovrun RW 0x0	bblerr RW 0x0	xacterr RW 0x0	nyet RW 0x0	ack RW 0x0	nak RW 0x0	stall RW 0x0	ahberr RW 0x0	chhltd RW 0x0	xfercompl RW 0x0
hcintmsk15 0x2EC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		frm_lst_rollintrmsk RW 0x0	Reserved	bnaintrmsk RW 0x0	datatglerrmsk RW 0x0	frmovrunmsk RW 0x0	bblerrmsk RW 0x0	xacterrmsk RW 0x0	nyetmsk RW 0x0	ackmsk RW 0x0	nakmsk RW 0x0	stallmsk RW 0x0	ahberrmsk RW 0x0	chhltdmsk RW 0x0	xfercomplmsk RW 0x0

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	dopng RW 0x0	pid RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
hctsiz15 0x2F0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	hcdma15 RW 0x0															
hcdma15 0x2F4	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hcdma15 RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	hcdmab15 RW 0x0															
hcdmab15 0x2FC	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	hcdmab15 RW 0x0															

hcfg

Host Configuration Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB00400
i_usbotg_1_hostgrp	0xFFB40400	0xFFB40400

Offset: 0x0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
modechtimen RW 0x0	Reserved				perschedena RW 0x0	frlisten RW 0x0		descdma RW 0x0	Reserved						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
resvalid RW 0x2								ena32khzs RW 0x0	Reserved				fslssupp RW 0x0	fslspclksel RW 0x0	

hcfg Fields

Bit	Name	Description	Access	Reset
31	modechtimen	Mode Change Ready Timer Enable (ModeChTimEn) This bit is used to enable/disable the Host core to wait 200 PHY clock cycles at the end of Resumeto change the opmode signal to the PHY to 00 after Suspend or LPM. 1'b0 : The Host core waits for either 200 PHY clock cycles or a linestate of SE0 at the end of resume to the change the opmode from 2'b10 to 2'b00 1'b1 : The Host core waits only for a linstate of SE0 at the end of resume to change the opmode from 2'b10 to 2'b00. **Value** **Description** 0x0 ENABLED 0x1 DISABLED	RW	0x0

Bit	Name	Description	Access	Reset
26	perschedena	Enable Periodic Scheduling (PerSchedEna): Applicable in host DDMA mode only. Enables periodic scheduling within the core. Initially, the bit is reset. The core will not process any periodic channels. As soon as this bit is set, the core will get ready to start scheduling periodic channels and sets HCFG.PerSchedStat. The setting of HCFG.PerSchedStat indicates the core has enabled periodic scheduling. Once HCFG.PerSchedEna is set, the application is not supposed to again reset the bit unless HCFG.PerSchedStat is set. As soon as this bit is reset, the core will get ready to stop scheduling periodic channels and resets HCFG.PerSchedStat. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
25:24	frlisten	Frame List Entries(FrListEn). The value in the register specifies the number of entries in the Frame list. This field is valid only in Scatter/Gather DMA mode. 2'b00: 8 Entries 2'b01: 16 Entries 2'b10: 32 Entries 2'b11: 63 Entries **Value** — **Description** 0x0 — RESERVED 0x1 — ENTRY8 0x2 — ENTRY16 0x3 — ENTRY32	RW	0x0

Bit	Name	Description	Access	Reset
23	descdma	Enable Scatter/gather DMA in Host mode (DescDMA). When the Scatter/Gather DMA option selected during configuration of the RTL, the application can set this bit during initialization to enable the Scatter/Gather DMA operation. NOTE: This bit must be modified only once after a reset. \The following combinations are available for programming: GAHBCFG.DMAEn=0,HCFG.DescDMA=0 => Slave mode GAHBCFG.DMAEn=0,HCFG.DescDMA=1 => Invalid GAHBCFG.DMAEn=1,HCFG.DescDMA=0 => Buffered DMA mode GAHBCFG.DMAEn=1,HCFG.DescDMA=1 => Scatter/Gather DMA mode **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
15:8	resvalid	Resume Validation Period (ResValid) This field is effective only when HCFG.Ena32KHzS is set. It will control the resume period when the core resumes from suspend. The core counts for 'ResValid' number of clock cycles to detect a valid resume when this is set.	RW	0x2

Bit	Name	Description	Access	Reset
7	ena32khzs	Enable 32 KHz Suspend mode (Ena32KHzS) This bit can be set only in FS PHY interface is selected. Else, this bit needs to be set to zero. When FS PHY interface is chosen and this bit is set, the core expects that the PHY clock during Suspend is switched from 48 MHz to 32 KHz. **Value** / **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
2	fslssupp	FS- and LS-Only Support (FSLSSupp) The application uses this bit to control the core's enumeration speed. Using this bit, the application can make the core enumerate as a FS host, even If the connected device supports HS traffic. Do not make changes to this field after initial programming. 1'b0: HS/FS/LS, based on the maximum speed supported by the connected device 1'b1: FS/LS-only, even If the connected device can support HS **Value** / **Description** 0x0 — HSFSLS 0x1 — FSLS	RW	0x0

Bit	Name	Description	Access	Reset
1:0	fslspclksel	FS/LS PHY Clock Select (FSLSPclkSel) When the core is in FS Host mode 2'b00: PHY clock is running at 30/60 MHz 2'b01: PHY clock is running at 48 MHz Others: Reserved When the core is in LS Host mode 2'b00: PHY clock is running at 30/60 MHz. When the UTMI+/ULPI PHY Low Power mode is not selected, use 30/60 MHz. 2'b01: PHY clock is running at 48 MHz. When the UTMI+ PHY Low Power mode is selected, use 48MHz If the PHY supplies a 48 MHz clock during LS mode. 2'b10: PHY clock is running at 6 MHz. In USB 1.1 FS mode, use 6 MHz when the UTMI+ PHY Low Power mode is selected and the PHY supplies a 6 MHz clock during LS mode. If you select a 6 MHz clock during LS mode, you must do a soft reset. 2'b11: Reserved Notes: When Core in FS mode, the internal and external clocks have the same frequency. When Core in LS mode, - If FSLSPclkSel = 2'b00: Internal and external clocks have the same frequency - If FSLSPclkSel = 2'b10: Internal clock is the divided by eight version of external 48 MHz clock **Value** — **Description** 0x0 — CLK3060 0x1 — CLK48 0x2 — CLK6	RW	0x0

hfir

Host Frame Interval Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB00404
i_usbotg_1_hostgrp	0xFFB40400	0xFFB40404

Offset: 0x4

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															hfirrldctrl RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
frint RW 0xEA60															

hfir Fields

Bit	Name	Description	Access	Reset
16	hfirrldctrl	Reload Control (HFIRRldCtrl) This bit allows dynamic reloading of the HFIR register during run time. 1'b0 : The HFIR cannot be reloaded dynamically 1'b1: the HFIR can be dynamically reloaded during runtime. This bit needs to be programmed during initial configuration and its value should not be changed during runtime. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
15:0	frint	Frame Interval (FrInt) The value that the application programs to this field specifies the interval between two consecutive SOFs (FS) or micro-SOFs (HS) or Keep-Alive tokens (HS). This field contains the number of PHY clocks that constitute the required frame interval. The Default value Set in this field For a FS operation when the PHY clock frequency is 60 MHz. The application can write a value to this register only after the Port Enable bit of the Host Port Control and Status register (HPRT.PrtEnaPort) has been Set. If no value is programmed, the core calculates the value based on the PHY clock specified in the FS/LS PHY Clock Select field of the Host Configuration register (HCFG.FSLSPclkSel). Do not change the value of this field after the initial configuration. 125 s * (PHY clock frequency For HS) 1 ms * (PHY clock frequency For FS/LS)	RW	0xEA60

hfnum

Host Frame Number/Frame Time Remaining Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB00408
i_usbotg_1_hostgrp	0xFFB40400	0xFFB40408

Offset: 0x8

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
frrem RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
frnum RO 0x3FFF															

hfnum Fields

Bit	Name	Description	Access	Reset
31:16	frrem	Frame Time Remaining (FrRem) Indicates the amount of time remaining in the current microframe (HS) or Frame (FS/LS), in terms of PHY clocks. This field decrements on each PHY clock. When it reaches zero, this field is reloaded with the value in the Frame Interval register and a new SOF is transmitted on the USB.	RO	0x0
15:0	frnum	Frame Number (FrNum) This field increments when a new SOF is transmitted on the USB, and is reset to 0 when it reaches 16'h3FFF. Value — Description 0x0 — INACTIVE 0x1 — ACTIVE	RO	0x3FFF

hptxsts

Host Periodic Transmit FIFO/Queue Status Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB00410
i_usbotg_1_hostgrp	0xFFB40400	0xFFB40410

Offset: 0x10

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
oddevnm-frame RO 0x0	chanendpt RO 0x0				type RO 0x0		term RO 0x0	ptxqspcavail RO 0x10							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ptxfspcavail RO 0x2000															

hptxsts Fields

Bit	Name	Description	Access	Reset
31	oddevnmframe	This indicates the odd/even micro frame that is currently being processes by the MAC. Value: Description 0x0: EVEN 0x1: ODD	RO	0x0

Bit	Name	Description	Access	Reset
30:27	chanendpt	This indicates the channel endpoint number that is currently being processes by the MAC. Value: Description 0xa: ENDPT10 0xb: ENDPT11 0xc: ENDPT12 0xd: ENDPT13 0xe: ENDPT14 0xf: ENDPT15 0x0: ENDPT0 0x1: ENDPT1 0x2: ENDPT2 0x3: ENDPT3 0x4: ENDPT4 0x5: ENDPT5 0x6: ENDPT6 0x7: ENDPT7 0x8: ENDPT8 0x9: ENDPT9	RO	0x0
26:25	type	This indicates the Entry in the Periodic Tx Request Queue that is currently being processes by the MAC. Value: Description 0x0: INOUT 0x1: ZEROLNGTH 0x2: CSPLIT 0x3: DISABLE	RO	0x0

Bit	Name	Description	Access	Reset
24	term	Terminate last entry for selected channel/endpoint. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0
23:16	ptxqspcavail	Periodic Transmit Request Queue Space Available (PTxQSpcAvail) Indicates the number of free locations available to be written in the Periodic Transmit Request Queue. This queue holds both IN and OUT requests. 8'h0: Periodic Transmit Request Queue is full 8'h1: 1 location available 8'h2: 2 locations available n: n locations available (0 <= n <= 16) Others: Reserved **Value** **Description** 0x0 FULL 0x1 FREE1 0x2 FREE2 0x3 FREE3 0x4 FREE4 0x5 FREE5 0x6 FREE6 0x7 FREE7 0x8 FREE8	RO	0x10

Bit	Name	Description	Access	Reset
15:0	ptxfspcavail	Periodic Transmit Data FIFO Space Available (PTxFSpcAvail) Indicates the number of free locations available to be written to in the Periodic TxFIFO. Values are in terms of 32-bit words 16'h0 : Periodic TxFIFO is full 16'h1 : 1 word available 16'h2 : 2 words available 16'hn : n words available (where 0 n 32,768) 16'h8000 : 32,768 words Others : Reserved	RO	0x2000

haint

Host All Channels Interrupt Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB00414
i_usbotg_1_hostgrp	0xFFB40400	0xFFB40414

Offset: 0x14

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
haint RO 0x0															

haint Fields

Bit	Name	Description	Access	Reset
15:0	haint	Channel Interrupt for channel no.	RO	0x0

haintmsk

Host All Channels Interrupt Mask Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB00418
i_usbotg_1_hostgrp	0xFFB40400	0xFFB40418

Offset: 0x18

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
haintmsk RW 0x0															

haintmsk Fields

Bit	Name	Description	Access	Reset
15:0	haintmsk	Channel Interrupt Msk for channel **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0

hflbaddr

Host Frame List Base Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB0041C
i_usbotg_1_hostgrp	0xFFB40400	0xFFB4041C

Offset: 0x1C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hflbaddr RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
hflbaddr RW 0x0															

hflbaddr Fields

Bit	Name	Description	Access	Reset
31:0	hflbaddr	The starting address of the Frame list. This register is used only for Isochronous and Interrupt Channels.	RW	0x0

hprt

Host Port Control and Status Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB00440
i_usbotg_1_hostgrp	0xFFB40400	0xFFB40440

Offset: 0x40

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													prtspd RO 0x0		prttstctl RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
prttstctl RW 0x0			prtpwr RW 0x0	prtlnsts RO 0x0		Reserved	prtrst RW 0x0	prtsusp RW 0x0	prtres RW 0x0	prtovrcurrchng RW 0x0	prtovrcurract RO 0x0	prtenchng RW 0x0	prtena RW 0x0	prtconndet RW 0x0	prtconnsts RO 0x0

hprt Fields

Bit	Name	Description	Access	Reset
18:17	prtspd	Port Speed (PrtSpd) Indicates the speed of the device attached to this port. 2'b00: High speed 2'b01: Full speed 2'b10: Low speed 2'b11: Reserved Value — Description 0x0 — HIGHSPD 0x1 — FULLSPD 0x2 — LOWSPD 0x3 — RESERVED	RO	0x0

Bit	Name	Description	Access	Reset
16:13	prttstctl	Port Test Control (PrtTstCtl) The application writes a nonzero value to this field to put the port into a Test mode, and the corresponding pattern is signaled on the port. 4'b0000: Test mode disabled 4'b0001: Test_J mode 4'b0010: Test_K mode 4'b0011: Test_SE0_NAK mode 4'b0100: Test_Packet mode 4'b0101: Test_Force_Enable Others: Reserved **Value** / **Description** 0x0 DISABLED 0x1 TESTJ 0x2 TESTK 0x3 TESTSN 0x4 TESTPM 0x5 TESTFENB	RW	0x0
12	prtpwr	Port Power (PrtPwr) The application uses this field to control power to this port (write 1'b1 to set to 1'b1 and write 1'b0 to set to 1'b0), and the core can clear this bit on an over current condition. 1'b0: Power off 1'b1: Power on **Value** / **Description** 0x0 OFF 0x1 ON	RW	0x0

Bit	Name	Description	Access	Reset
11:10	prtlnsts	Port Line Status (PrtLnSts) Indicates the current logic level USB data lines Bit [10]: Logic level of D+ Bit [11]: Logic level of D- **Value** / **Description** 0x1 — PLUSD 0x2 — MINUSD	RO	0x0
8	prtrst	Port Reset (PrtRst) When the application sets this bit, a reset sequence is started on this port. The application must time the reset period and clear this bit after the reset sequence is complete. 1'b0: Port not in reset 1'b1: Port in reset The application must leave this bit Set For at least a minimum duration mentioned below to start a reset on the port. The application can leave it Set For another 10 ms in addition to the required minimum duration, before clearing the bit, even though there is no maximum limit Set by the USB standard.This bit is cleared by the core even if there is no device connected to the Host. High speed: 50 ms Full speed/Low speed: 10 ms **Value** / **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
7	prtsusp	Port Suspend (PrtSusp) The application sets this bit to put this port in Suspend mode. The core only stops sending SOFs when this is Set. To stop the PHY clock, the application must Set the Port Clock Stop bit, which asserts the suspend input pin of the PHY. The read value of this bit reflects the current suspend status of the port. This bit is cleared by the core after a remote wakeup signal is detected or the application sets the Port Reset bit or Port Resume bit in this register or the Resume/Remote Wakeup Detected Interrupt bit or Disconnect Detected Interrupt bit in the Core Interrupt register (GINTSTS.WkUpInt or GINTSTS.DisconnInt, respectively).This bit is cleared by the core even if there is no device connected to the Host. 1'b0: Port not in Suspend mode 1'b1: Port in Suspend mode **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
6	prtres	Port Resume (PrtRes) The application sets this bit to drive resume signaling on the port. The core continues to drive the resume signal until the application clears this bit. If the core detects a USB remote wakeup sequence, as indicated by the Port Resume/Remote Wakeup Detected Interrupt bit of the Core Interrupt register (GINTSTS.WkUpInt), the core starts driving resume signaling without application intervention and clears this bit when it detects a disconnect condition. The read value of this bit indicates whether the core is currently driving resume signaling. 1'b0: No resume driven 1'b1: Resume driven When LPM is enabled, In L1 state the behavior of this bit is as follows: The application sets this bit to drive resume signaling on the port. The core continues to drive the resume signal until a pre-determined time specified in GLPMCFG.HIRD_Thres[3:0] field. If the core detects a USB remote wakeup sequence, as indicated by the Port L1Resume/Remote L1Wakeup Detected Interrupt bit of the Core Interrupt register (GINTSTS.L1WkUpInt), the core starts driving resume signaling without application intervention and clears this bit at the end of resume.This bit can be set by both core or application and also cleared by core or application. This bit is cleared by the core even if there is no device connected to the Host. **Value** — **Description** 0x0 — NORESUME 0x1 — RESUME	RW	0x0

Bit	Name	Description	Access	Reset
5	prtovrcurrchng	Port Overcurrent Change (PrtOvrCurrChng) The core sets this bit when the status of the Port Overcurrent Active bit (bit 4) in this register changes.This bit can be set only by the core and the application should write 1 to clear it **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
4	prtovrcurract	Port Overcurrent Active (PrtOvrCur-rAct) Indicates the overcurrent condition of the port. 1'b0: No overcurrent condition 1'b1: Overcurrent condition **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0
3	prtenchng	Port Enable/Disable Change (PrtEnChng) The core sets this bit when the status of the Port Enable bit [2] of this register changes.This bit can be set only by the core and the application should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
2	prtena	Port Enable (PrtEna) A port is enabled only by the core after a reset sequence, and is disabled by an overcurrent condition, a disconnect condition, or by the application clearing this bit. The application cannot Set this bit by a register write. It can only clear it to disable the port by writing 1.. This bit does not trigger any interrupt to the application. 1'b0: Port disabled 1'b1: Port enabled **Value** / **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
1	prtconndet	Port Connect Detected (PrtConnDet) The core sets this bit when a device connection is detected to trigger an interrupt to the application using the Host Port Interrupt bit of the Core Interrupt register (GINTSTS.PrtInt).This bit can be set only by the core and the application should write 1 to clear it.The application must write a 1 to this bit to clear the interrupt. **Value** / **Description** 0x0 ACTIVE 0x1 INACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
0	prtconnsts	Port Connect Status (PrtConnSts) 0: No device is attached to the port. 1: A device is attached to the port. Value: 0x0 Description: NOTATTACHED Value: 0x1 Description: ATTACHED	RO	0x0

hcchar0

Host Channel 0 Characteristics Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB00500
i_usbotg_1_hostgrp	0xFFB40400	0xFFB40500

Offset: 0x100

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
chena RW 0x0	chdis RW 0x0	oddfrm RW 0x0	devaddr RW 0x0							ec RW 0x0		eptype RW 0x0		lspddev RW 0x0	Reserved
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
epdir RW 0x0	epnum RW 0x0				mps RW 0x0										

hcchar0 Fields

Bit	Name	Description	Access	Reset
31	chena	Channel Enable (ChEna) When Scatter/Gather mode is enabled 1'b0: Indicates that the descriptor structure is not yet ready. 1'b1: Indicates that the descriptor structure and data buffer with data is setup and this channel can access the descriptor. When Scatter/Gather mode is disabled This field is set by the applica-tion and cleared by the OTG host. 1'b0: Channel disabled 1'b1: Channel enabled **Value** **Description** 0x0 NOTREADY 0x1 READY	RW	0x0
30	chdis	Channel Disable (ChDis) The application sets this bit to stop transmitting/receiving data on a channel, even before the transfer For that channel is complete. The application must wait For the Channel Disabled interrupt before treating the channel as disabled. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
29	oddfrm	Odd Frame (OddFrm) This field is set (reset) by the application to indicate that the OTG host must perform a transfer in an odd (micro)frame. This field is applicable for only periodic (isochronous and interrupt) transac-tions. 1'b0: Even (micro)frame 1'b1: Odd (micro)frame	RW	0x0
28:22	devaddr	Device Address (DevAddr) This field selects the specific device serving as the data source or sink.	RW	0x0

Bit	Name	Description	Access	Reset
21:20	ec	Multi Count (MC) / Error Count (EC) When the Split Enable bit of the Host Channel-n Split Control register (HCSPLTn.SpltEna) is reset (1'b0), this field indicates to the host the number of transactions that must be executed per microframe For this periodic endpoint. For non periodic transfers, this field is used only in DMA mode, and specifies the number packets to be fetched For this channel before the internal DMA engine changes arbitration. 2'b00: Reserved This field yields undefined results. 2'b01: 1 transaction 2'b10: 2 transactions to be issued For this endpoint per microframe 2'b11: 3 transactions to be issued For this endpoint per microframe When HCSPLTn.SpltEna is Set (1'b1), this field indicates the number of immediate retries to be performed For a periodic split transactions on transaction errors. This field must be Set to at least 2'b01. **Value** **Description** 0x0 RESERVED 0x1 TRANSONE 0x2 TRANSTWO 0x3 TRANSTHREE	RW	0x0

Bit	Name	Description	Access	Reset
19:18	eptype	Endpoint Type (EPType) Indicates the transfer type selected. 2'b00: Control 2'b01: Isochronous 2'b10: Bulk 2'b11: Interrupt **Value** — **Description** 0x0 — CTRL 0x1 — ISOC 0x2 — BULK 0x3 — INTERR	RW	0x0
17	lspddev	Low-Speed Device (LSpdDev) This field is Set by the application to indicate that this channel is communicating to a low-speed device. **Value** — **Description** 0x0 — NONLOWSPEED 0x1 — LOWSPEED	RW	0x0
15	epdir	Endpoint Direction (EPDir) Indicates whether the transaction is IN or OUT. 1'b0: OUT 1'b1: IN **Value** — **Description** 0x0 — OUTDIR 0x1 — INDIR	RW	0x0

Bit	Name	Description	Access	Reset
14:11	epnum	Endpoint Number (EPNum) Indicates the endpoint number on the device serving as the data source or sink. Value — Description 0xa — ENDPT10 0xb — ENDPT11 0xc — ENDPT12 0xd — ENDPT13 0xe — ENDPT14 0xf — ENDPT15 0x0 — ENDPT0 0x1 — ENDPT1 0x2 — ENDPT2 0x3 — ENDPT3 0x4 — ENDPT4 0x5 — ENDPT5 0x6 — ENDPT6 0x7 — ENDPT7 0x8 — ENDPT8 0x9 — ENDPT9	RW	0x0
10:0	mps	Maximum Packet Size (MPS) Indicates the maximum packet size of the associated endpoint.	RW	0x0

hcsplt0

Host Channel 0 Split Control Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB00504
i_usbotg_1_hostgrp	0xFFB40400	0xFFB40504

Offset: 0x104

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
spltena RW 0x0	Reserved														compsplt RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
xactpos RW 0x0	hubaddr RW 0x0								prtaddr RW 0x0						

hcsplt0 Fields

Bit	Name	Description	Access	Reset
31	spltena	Split Enable (SpltEna) The application sets this field to indicate that this channel is enabled to perform split transactions. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
16	compsplt	Do Complete Split (CompSplt) The application sets this field to request the OTG host to perform a complete split transaction. **Value** — **Description** 0x0 — NOSPLIT 0x1 — SPLIT	RW	0x0

Bit	Name	Description	Access	Reset
15:14	xactpos	Transaction Position (XactPos) This field is used to determine whether to send all, first, middle, or last payloads with each OUT transaction. 2'b11: All. This is the entire data payload is of this transaction (which is less than or equal to 188 bytes). 2'b10: Begin. This is the first data payload of this transaction (which is larger than 188 bytes). 2'b00: Mid. This is the middle payload of this transaction (which is larger than 188 bytes). 2'b01: End. This is the last payload of this transaction (which is larger than 188 bytes). **Value** **Description** 0x0 MIDDLE 0x1 END 0x2 BEGIN 0x3 ALL	RW	0x0
13:7	hubaddr	Hub Address (HubAddr) This field holds the device address of the transaction translator's hub.	RW	0x0
6:0	prtaddr	Port Address (PrtAddr) This field is the port number of the recipient transaction translator.	RW	0x0

hcint0

Host Channel 0 Interrupt Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB00508

Module Instance	Base Address	Register Address
i_usbotg_1_hostgrp	0xFFB40400	0xFFB40508

Offset: 0x108

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		desc_lst_rollintr RW 0x0	xcs_xact_err RW 0x0	bnaintr RW 0x0	datatglerr RW 0x0	frmovrun RW 0x0	bblerr RW 0x0	xacterr RW 0x0	nyet RW 0x0	ack RW 0x0	nak RW 0x0	stall RW 0x0	ahberr RW 0x0	chhltd RW 0x0	xfercompl RW 0x0

hcint0 Fields

Bit	Name	Description	Access	Reset
13	desc_lst_rollintr	Descriptor rollover interrupt (DESC_LST_ROLLIntr) This bit is valid only when Scatter/Gather DMA mode is enabled. The core sets this bit when the corresponding channel's descriptor list rolls over. For non Scatter/Gather DMA mode, this bit is reserved. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
12	xcs_xact_err	Excessive Transaction Error (XCS_XACT_ERR) This bit is valid only when Scatter/Gather DMA mode is enabled. The core sets this bit when 3 consecutive transaction errors occurred on the USB bus. XCS_XACT_ERR will not be generated for Isochronous channels. For non Scatter/Gather DMA mode, this bit is reserved. **Value** **Description** 0x0 INACTIVE 0x1 ACVTIVE	RW	0x0
11	bnaintr	BNA (Buffer Not Available) Interrupt (BNAIntr) This bit is valid only when Scatter/Gather DMA mode is enabled. The core generates this interrupt when the descriptor accessed is not ready for the Core to process. BNA will not be generated for Isochronous channels. For non Scatter/Gather DMA mode, this bit is reserved. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
10	datatglerr	Data Toggle Error (DataTglErr).This bit can be set only by the core and the application should write 1 to clear it.In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
9	frmovrun	Frame Overrun (FrmOvrun).In Scatter/ Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the applica- tion should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
8	bblerr	Babble Error (BblErr) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core..This bit can be set only by the core and the applica- tion should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
7	xacterr	Transaction Error (XactErr) Indicates one of the following errors occurred on the USB. CRC check failure Timeout Bit stuff error False EOP In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the applica- tion should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
6	nyet	NYET Response Received Interrupt (NYET) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
5	ack	ACK Response Received/Transmitted Interrupt (ACK) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
4	nak	NAK Response Received Interrupt (NAK) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
3	stall	STALL Response Received Interrupt (STALL) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value** / **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
2	ahberr	AHB Error (AHBErr) This is generated only in Internal DMA mode when there is an AHB error during AHB read/write. The application can read the corresponding channel's DMA address register to get the error address. **Value** / **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
1	chhltd	Channel Halted (ChHltd) In non Scatter/Gather DMA mode, it indicates the transfer completed abnormally either because of any USB transaction error or in response to disable request by the application or because of a completed transfer. in Scatter/gather DMA mode, this indicates that transfer completed due to any of the following . EOL being set in descriptor . AHB error . Excessive transaction errors . Babble . Stall **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
0	xfercompl	Transfer Completed (XferCompl) Transfer completed normally without any errors.This bit can be set only by the core and the application should write 1 to clear it. For Scatter/Gather DMA mode, it indicates that current descriptor processing got completed with IOC bit set in its descriptor. In non Scatter/Gather DMA mode, it indicates that Transfer completed normally without any errors. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

hcintmsk0

Host Channel 0 Interrupt Mask Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB0050C
i_usbotg_1_hostgrp	0xFFB40400	0xFFB4050C

Offset: 0x10C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		frm_lst_rollintrmsk RW 0x0	Reserved	bnaintrmsk RW 0x0	datatglerrmsk RW 0x0	frmovrunmsk RW 0x0	bblerrmsk RW 0x0	xacterrmsk RW 0x0	nyetmsk RW 0x0	ackmsk RW 0x0	nakmsk RW 0x0	stallmsk RW 0x0	ahberrmsk RW 0x0	chhltdmsk RW 0x0	xfercomplmsk RW 0x0

hcintmsk0 Fields

Bit	Name	Description	Access	Reset
13	frm_lst_rollintrmsk	Framelist rollover interrupt Mask register(FRM_LST_ROLLIntrMsk) This bit is valid only when Scatter/Gather DMA mode is enabled. Value: Description 0x0: MASK 0x1: NOMASK	RW	0x0

Bit	Name	Description	Access	Reset
11	bnaintrmsk	BNA (Buffer Not Available) Interrupt mask register (BNAIntrMsk) This bit is valid only when Scatter/Gather DMA mode is enabled. Value: 0x0 Description: MASK Value: 0x1 Description: NOMASK	RW	0x0
10	datatglerrmsk	Data Toggle Error Mask (DataTglErrMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
9	frmovrunmsk	Frame Overrun Mask (FrmOvrunMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
8	bblerrmsk	Babble Error Mask (BblErrMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
7	xacterrmsk	Transaction Error Mask (XactErrMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0

Bit	Name	Description	Access	Reset
6	nyetmsk	NYET Response Received Interrupt Mask (NyetMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
5	ackmsk	ACK Response Received/Transmitted Interrupt Mask (AckMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
4	nakmsk	NAK Response Received Interrupt Mask (NakMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
3	stallmsk	STALL Response Received Interrupt Mask (StallMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0

Bit	Name	Description	Access	Reset
2	ahberrmsk	AHB Error Mask (AHBErrMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn. **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0
1	chhltdmsk	Channel Halted Mask (ChHltdMsk) **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0
0	xfercomplmsk	Transfer Completed Mask (XferComplMsk) **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0

hctsiz0

Host Channel 0 Transfer Size Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB00510
i_usbotg_1_hostgrp	0xFFB40400	0xFFB40510

Offset: 0x110

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dopng RW 0x0	pid RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
xfersize RW 0x0															

hctsiz0 Fields

Bit	Name	Description	Access	Reset
31	dopng	Do Ping (DoPng) This bit is used only For OUT transfers. Setting this field to 1 directs the host to do PING protocol. Note: Do not Set this bit For IN transfers. If this bit is Set For for IN transfers it disables the channel. **Value** — **Description** 0x0 — NOPING 0x1 — PING	RW	0x0

Bit	Name	Description	Access	Reset
30:29	pid	PID (Pid) The application programs this field with the type of PID to use For the initial transaction. The host maintains this field For the rest of the transfer. 2'b00: DATA0 2'b01: DATA2 2'b10: DATA1 2'b11: MDATA (non-control)/SETUP (control) **Value** **Description** 0x0 DATA0 0x1 DATA2 0x2 DATA1 0x3 MDATA	RW	0x0
28:19	pktcnt	Packet Count (PktCnt) This field is programmed by the application with the expected number of packets to be transmitted (OUT) or received (IN). The host decrements this count on every successful transmission or reception of an OUT/IN packet. Once this count reaches zero, the application is interrupted to indicate normal completion. The width of this counter is specified as Width of Packet Counters	RW	0x0
18:0	xfersize	Transfer Size (XferSize) For an OUT, this field is the number of data bytes the host sends during the transfer. For an IN, this field is the buffer size that the application has Reserved For the transfer. The application is expected to program this field as an integer multiple of the maximum packet size For IN transactions (periodic and non-periodic). The width of this counter is specified as Width of Transfer Size Counters	RW	0x0

hcdma0

Host Channel 0 DMA Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB00514
i_usbotg_1_hostgrp	0xFFB40400	0xFFB40514

Offset: 0x114

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcdma0 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
hcdma0 RW 0x0															

hcdma0 Fields

Bit	Name	Description	Access	Reset
31:0	hcdma0	Buffer DMA Mode: [31:0] DMA Address (DMAAddr) This field holds the start address in the external memory from which the data for the endpoint must be fetched or to which it must be stored. This register is incremented on every AHB transaction. Scatter-Gather DMA (DescDMA) Mode: [31:9] (Non Isoc) Non-Isochronous: [31:N] (Isoc) Isochronous: This field holds the start address of the 512 bytes page. The first descriptor in the list should be located in this address. The first descriptor may be or may not be ready. The core starts processing the list from the CTD value.	RW	0x0

Bit	Name	Description	Access	Reset
		This field holds the address of the 2*(nTD+1) bytes of locations in which the isochronous descriptors are present where N is based on nTD as per Table below [31:N] Base Address [N-1:3] Offset [2:0] 000 HS ISOC nTD N 7 6 15 7 31 8 63 9 127 10 255 11 FS ISOC nTD N 1 4 3 5 7 6 15 7 31 8 63 9 [N-1:3] (Isoc): [8:3] (Non Isoc): Current Transfer Desc(CTD): Non Isochronous: This value is in terms of number of descriptors. The values can be from 0 to 63. • 0 - 1 descriptor. • 63 - 64 descriptors. This field indicates the current descriptor processed in the list. This field is updated both by application and the core. For example, if the application enables the channel after programming CTD=5, then the core will start processing the 6th descriptor. The address is obtained by adding a value of (8bytes*5=) 40(decimal) to DMAAddr.		

Bit	Name	Description	Access	Reset
		Isochronous: CTD for isochronous is based on the current frame/ (micro)frame value. Need to be set to zero by application.Scatter-Gather DMA (DescDMA) Mode: [31:9] (Non Isoc) Non-Isochronous: [31:N] (Isoc) Isochronous: This field holds the start address of the 512 bytes page. The first descriptor in the list should be located in this address. The first descriptor may be or may not be ready. The core starts processing the list from the CTD value. This field holds the address of the 2*(nTD+1) bytes of locations in which the isochronous descriptors are present where N is based on nTD as per Table below [31:N] Base Address [N-1:3] Offset [2:0] 000 HS ISOC nTD N 7 6 15 7 31 8 63 9 127 10 255 11 FS ISOC nTD N 1 4 3 5 7 6 15 7 31 8 63 9		

Bit	Name	Description	Access	Reset
		[N-1:3] (Isoc): [8:3] (Non Isoc): Current Transfer Desc(CTD): Non Isochronous: This value is in terms of number of descriptors. The values can be from 0 to 63. • 0 - 1 descriptor. • 63 - 64 descriptors. This field indicates the current descriptor processed in the list. This field is updated both by application and the core. For example, if the application enables the channel after programming CTD=5, then the core will start processing the 6th descriptor. The address is obtained by adding a value of (8bytes*5=) 40(decimal) to DMAAddr. Isochronous: CTD for isochronous is based on the current frame/ (micro)frame value. Need to be set to zero by application.		

hcdmab0

Host Channel 0 DMA Buffer Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB0051C
i_usbotg_1_hostgrp	0xFFB40400	0xFFB4051C

Offset: 0x11C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcdmab0 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
hcdmab0 RW 0x0															

hcdmab0 Fields

Bit	Name	Description	Access	Reset
31:0	hcdmab0	Holds the current buffer address. This register is updated as and when the data transfer for the corresponding end point is in progress. This register is present only in Scatter/Gather DMA mode. Otherwise this field is reserved.	RW	0x0

hcchar1

Host Channel 1 Characteristics Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB00520
i_usbotg_1_hostgrp	0xFFB40400	0xFFB40520

Offset: 0x120

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
chena RW 0x0	chdis RW 0x0	oddfrm RW 0x0	devaddr RW 0x0							ec RW 0x0		eptype RW 0x0		lspddev RW 0x0	Reserved
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
epdir RW 0x0	epnum RW 0x0				mps RW 0x0										

hcchar1 Fields

Bit	Name	Description	Access	Reset
31	chena	Channel Enable (ChEna) When Scatter/Gather mode is enabled 1'b0: Indicates that the descriptor structure is not yet ready. 1'b1: Indicates that the descriptor structure and data buffer with data is setup and this channel can access the descriptor. When Scatter/Gather mode is disabled This field is set by the application and cleared by the OTG host. 1'b0: Channel disabled 1'b1: Channel enabled **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
30	chdis	Channel Disable (ChDis) The application sets this bit to stop transmitting/receiving data on a channel, even before the transfer For that channel is complete. The application must wait For the Channel Disabled interrupt before treating the channel as disabled. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
29	oddfrm	Odd Frame (OddFrm) This field is set (reset) by the application to indicate that the OTG host must perform a transfer in an odd (micro)frame. This field is applicable for only periodic (isochronous and interrupt) transactions. 1'b0: Even (micro)frame 1'b1: Odd (micro)frame	RW	0x0
28:22	devaddr	Device Address (DevAddr) This field selects the specific device serving as the data source or sink.	RW	0x0

Bit	Name	Description	Access	Reset
21:20	ec	Multi Count (MC) / Error Count (EC) When the Split Enable bit of the Host Channel-n Split Control register (HCSPLTn.SpltEna) is reset (1'b0), this field indicates to the host the number of transactions that must be executed per microframe For this periodic endpoint. For non periodic transfers, this field is used only in DMA mode, and specifies the number packets to be fetched For this channel before the internal DMA engine changes arbitration. 2'b00: Reserved This field yields undefined results. 2'b01: 1 transaction 2'b10: 2 transactions to be issued For this endpoint per microframe 2'b11: 3 transactions to be issued For this endpoint per microframe When HCSPLTn.SpltEna is Set (1'b1), this field indicates the number of immediate retries to be performed For a periodic split transactions on transaction errors. This field must be Set to at least 2'b01. **Value** **Description** 0x0 RESERVED 0x1 TRANSONE 0x2 TRANSTWO 0x3 TRANSTHREE	RW	0x0

Bit	Name	Description	Access	Reset
19:18	eptype	Endpoint Type (EPType) Indicates the transfer type selected. 2'b00: Control 2'b01: Isochronous 2'b10: Bulk 2'b11: Interrupt **Value** — **Description** 0x0 — CTRL 0x1 — ISOC 0x2 — BULK 0x3 — INTERR	RW	0x0
17	lspddev	Low-Speed Device (LSpdDev) This field is Set by the application to indicate that this channel is communicating to a low-speed device. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
15	epdir	Endpoint Direction (EPDir) Indicates whether the transaction is IN or OUT. 1'b0: OUT 1'b1: IN **Value** — **Description** 0x0 — OUT 0x1 — IN	RW	0x0

Bit	Name	Description	Access	Reset
14:11	epnum	Endpoint Number (EPNum) Indicates the endpoint number on the device serving as the data source or sink. Value — Description 0xa — ENDPT10 0xb — ENDPT11 0xc — ENDPT12 0xd — ENDPT13 0xe — ENDPT14 0xf — ENDPT15 0x0 — ENDPT0 0x1 — ENDPT1 0x2 — ENDPT2 0x3 — ENDPT3 0x4 — ENDPT4 0x5 — ENDPT5 0x6 — ENDPT6 0x7 — ENDPT7 0x8 — ENDPT8 0x9 — ENDPT9	RW	0x0
10:0	mps	Maximum Packet Size (MPS) Indicates the maximum packet size of the associated endpoint.	RW	0x0

hcsplt1

Host Channel 1 Split Control Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB00524
i_usbotg_1_hostgrp	0xFFB40400	0xFFB40524

Offset: 0x124

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
spltena RW 0x0	Reserved														compsplt RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
xactpos RW 0x0	hubaddr RW 0x0								prtaddr RW 0x0						

hcsplt1 Fields

Bit	Name	Description	Access	Reset
31	spltena	Split Enable (SpltEna) The application sets this field to indicate that this channel is enabled to perform split transactions. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
16	compsplt	Do Complete Split (CompSplt) The application sets this field to request the OTG host to perform a complete split transaction. **Value** — **Description** 0x0 — NOSPLIT 0x1 — SPLIT	RW	0x0

Bit	Name	Description	Access	Reset
15:14	xactpos	Transaction Position (XactPos) This field is used to determine whether to send all, first, middle, or last payloads with each OUT transaction. 2'b11: All. This is the entire data payload is of this transaction (which is less than or equal to 188 bytes). 2'b10: Begin. This is the first data payload of this transaction (which is larger than 188 bytes). 2'b00: Mid. This is the middle payload of this transaction (which is larger than 188 bytes). 2'b01: End. This is the last payload of this transaction (which is larger than 188 bytes). **Value** **Description** 0x0 MIDDLE 0x1 END 0x2 BEGIN 0x3 ALL	RW	0x0
13:7	hubaddr	Hub Address (HubAddr) This field holds the device address of the transaction translator's hub.	RW	0x0
6:0	prtaddr	Port Address (PrtAddr) This field is the port number of the recipient transaction translator.	RW	0x0

hcint1

Host Channel 1 Interrupt Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB00528

Module Instance	Base Address	Register Address
i_usbotg_1_hostgrp	0xFFB40400	0xFFB40528

Offset: 0x128

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		desc_lst_rollintr RW 0x0	xcs_xact_err RW 0x0	bnaintr RW 0x0	datatglerr RW 0x0	frmovrun RW 0x0	bblerr RW 0x0	xacterr RW 0x0	nyet RW 0x0	ack RW 0x0	nak RW 0x0	stall RW 0x0	ahberr RW 0x0	chhltd RW 0x0	xfercompl RW 0x0

hcint1 Fields

Bit	Name	Description	Access	Reset
13	desc_lst_rollintr	Descriptor rollover interrupt (DESC_LST_ROLLIntr) This bit is valid only when Scatter/Gather DMA mode is enabled. The core sets this bit when the corresponding channel's descriptor list rolls over. For non Scatter/Gather DMA mode, this bit is reserved. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
12	xcs_xact_err	Excessive Transaction Error (XCS_XACT_ERR) This bit is valid only when Scatter/Gather DMA mode is enabled. The core sets this bit when 3 consecutive transaction errors occurred on the USB bus. XCS_XACT_ERR will not be generated for Isochronous channels. For non Scatter/Gather DMA mode, this bit is reserved. **Value** **Description** 0x0 INACTIVE 0x1 ACVTIVE	RW	0x0
11	bnaintr	BNA (Buffer Not Available) Interrupt (BNAIntr) This bit is valid only when Scatter/Gather DMA mode is enabled. The core generates this interrupt when the descriptor accessed is not ready for the Core to process. BNA will not be generated for Isochronous channels. For non Scatter/Gather DMA mode, this bit is reserved. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
10	datatglerr	Data Toggle Error (DataTglErr).This bit can be set only by the core and the application should write 1 to clear it.In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
9	frmovrun	Frame Overrun (FrmOvrun).In Scatter/ Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the applica- tion should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
8	bblerr	Babble Error (BblErr) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core..This bit can be set only by the core and the applica- tion should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
7	xacterr	Transaction Error (XactErr) Indicates one of the following errors occurred on the USB. CRC check failure Timeout Bit stuff error False EOP In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the applica- tion should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
6	nyet	NYET Response Received Interrupt (NYET) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
5	ack	ACK Response Received/Transmitted Interrupt (ACK) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
4	nak	NAK Response Received Interrupt (NAK) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
3	stall	STALL Response Received Interrupt (STALL) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value** / **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
2	ahberr	AHB Error (AHBErr) This is generated only in Internal DMA mode when there is an AHB error during AHB read/write. The application can read the corresponding channel's DMA address register to get the error address. **Value** / **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
1	chhltd	Channel Halted (ChHltd) In non Scatter/Gather DMA mode, it indicates the transfer completed abnormally either because of any USB transaction error or in response to disable request by the application or because of a completed transfer. in Scatter/gather DMA mode, this indicates that transfer completed due to any of the following . EOL being set in descriptor . AHB error . Excessive transaction errors . Babble . Stall **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
0	xfercompl	Transfer Completed (XferCompl) Transfer completed normally without any errors.This bit can be set only by the core and the application should write 1 to clear it. For Scatter/Gather DMA mode, it indicates that current descriptor processing got completed with IOC bit set in its descriptor. In non Scatter/Gather DMA mode, it indicates that Transfer completed normally without any errors. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

hcintmsk1

Host Channel 1 Interrupt Mask Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB0052C
i_usbotg_1_hostgrp	0xFFB40400	0xFFB4052C

Offset: 0x12C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		frm_lst_rollintrmsk RW 0x0	Reserved	bnaintrmsk RW 0x0	datatglerrmsk RW 0x0	frmovrunmsk RW 0x0	bblerrmsk RW 0x0	xacterrmsk RW 0x0	nyetmsk RW 0x0	ackmsk RW 0x0	nakmsk RW 0x0	stallmsk RW 0x0	ahberrmsk RW 0x0	chhltdmsk RW 0x0	xfercomplmsk RW 0x0

hcintmsk1 Fields

Bit	Name	Description	Access	Reset
13	frm_lst_rollintrmsk	Framelist rollover interrupt Mask register(FRM_LST_ROLLIntrMsk) This bit is valid only when Scatter/Gather DMA mode is enabled. Value: 0x0 — MASK; 0x1 — NOMASK	RW	0x0

Bit	Name	Description	Access	Reset
11	bnaintrmsk	BNA (Buffer Not Available) Interrupt mask register (BNAIntrMsk) This bit is valid only when Scatter/Gather DMA mode is enabled. Value: 0x0 Description: MASK Value: 0x1 Description: NOMASK	RW	0x0
10	datatglerrmsk	Data Toggle Error Mask (DataTglErrMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
9	frmovrunmsk	Frame Overrun Mask (FrmOvrunMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
8	bblerrmsk	Babble Error Mask (BblErrMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
7	xacterrmsk	Transaction Error Mask (XactErrMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0

Bit	Name	Description	Access	Reset
6	nyetmsk	NYET Response Received Interrupt Mask (NyetMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
5	ackmsk	ACK Response Received/Transmitted Interrupt Mask (AckMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
4	nakmsk	NAK Response Received Interrupt Mask (NakMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
3	stallmsk	STALL Response Received Interrupt Mask (StallMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0

Bit	Name	Description	Access	Reset
2	ahberrmsk	AHB Error Mask (AHBErrMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn. **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0
1	chhltdmsk	Channel Halted Mask (ChHltdMsk) **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0
0	xfercomplmsk	Transfer Completed Mask (XferComplMsk) **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0

hctsiz1

Host Channel 1 Transfer Size Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB00530
i_usbotg_1_hostgrp	0xFFB40400	0xFFB40530

Offset: 0x130

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dopng RW 0x0	pid RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
xfersize RW 0x0															

hctsiz1 Fields

Bit	Name	Description	Access	Reset
31	dopng	Do Ping (DoPng) This bit is used only For OUT transfers. Setting this field to 1 directs the host to do PING protocol. Note: Do not Set this bit For IN transfers. If this bit is Set For for IN transfers it disables the channel. **Value** — **Description** 0x0 — NOPING 0x1 — PING	RW	0x0

Bit	Name	Description	Access	Reset
30:29	pid	PID (Pid) The application programs this field with the type of PID to use For the initial transaction. The host maintains this field For the rest of the transfer. 2'b00: DATA0 2'b01: DATA2 2'b10: DATA1 2'b11: MDATA (non-control)/SETUP (control) **Value** **Description** 0x0 DATA0 0x1 DATA2 0x2 DATA1 0x3 MDATA	RW	0x0
28:19	pktcnt	Packet Count (PktCnt) This field is programmed by the application with the expected number of packets to be transmitted (OUT) or received (IN). The host decrements this count on every successful transmission or reception of an OUT/IN packet. Once this count reaches zero, the application is interrupted to indicate normal completion. The width of this counter is specified as Width of Packet Counters	RW	0x0
18:0	xfersize	Transfer Size (XferSize) For an OUT, this field is the number of data bytes the host sends during the transfer. For an IN, this field is the buffer size that the application has Reserved For the transfer. The application is expected to program this field as an integer multiple of the maximum packet size For IN transactions (periodic and non-periodic). The width of this counter is specified as Width of Transfer Size Counters	RW	0x0

hcdma1

Host Channel 1 DMA Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB00534
i_usbotg_1_hostgrp	0xFFB40400	0xFFB40534

Offset: 0x134

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcdma1 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
hcdma1 RW 0x0															

hcdma1 Fields

Bit	Name	Description	Access	Reset
31:0	hcdma1	Buffer DMA Mode: [31:0] DMA Address (DMAAddr) This field holds the start address in the external memory from which the data for the endpoint must be fetched or to which it must be stored. This register is incremented on every AHB transac-tion. Scatter-Gather DMA (DescDMA) Mode: [31:9] (Non Isoc) Non-Isochronous: [31:N] (Isoc) Isochronous: This field holds the start address of the 512 bytes page. The first descriptor in the list should be located in this address. The first	RW	0x0

Bit	Name	Description	Access	Reset
		descriptor may be or may not be ready. The core starts processing the list from the CTD value. This field holds the address of the 2*(nTD+1) bytes of locations in which the isochronous descriptors are present where N is based on nTD as per Table below [31:N] Base Address [N-1:3] Offset [2:0] 000 HS ISOC nTD N 7 6 15 7 31 8 63 9 127 10 255 11 FS ISOC nTD N 1 4 3 5 7 6 15 7 31 8 63 9 [N-1:3] (Isoc): [8:3] (Non Isoc): Current Transfer Desc(CTD): Non Isochronous: This value is in terms of number of descriptors. The values can be from 0 to 63. 0 - 1 descriptor. 63 - 64 descriptors. This field indicates the current descriptor processed in the list. This field is updated both by application and the core. For example, if the application enables the channel after programming CTD=5, then the core will start processing the 6th descriptor. The address is obtained		

Bit	Name	Description	Access	Reset
		by adding a value of (8bytes*5=) 40(decimal) to DMAAddr. Isochronous: CTD for isochronous is based on the current frame/(micro)frame value. Need to be set to zero by application.Scatter-Gather DMA (DescDMA) Mode: [31:9] (Non Isoc) Non-Isochronous: [31:N] (Isoc) Isochronous: This field holds the start address of the 512 bytes page. The first descriptor in the list should be located in this address. The first descriptor may be or may not be ready. The core starts processing the list from the CTD value. This field holds the address of the 2*(nTD+1) bytes of locations in which the isochronous descriptors are present where N is based on nTD as per Table below [31:N] Base Address [N-1:3] Offset [2:0] 000 HS ISOC nTD N 7 6 15 7 31 8 63 9 127 10 255 11 FS ISOC nTD N 1 4 3 5 7 6 15 7 31 8 63 9 [N-1:3] (Isoc): [8:3] (Non Isoc): Current Transfer Desc(CTD):		

Bit	Name	Description	Access	Reset
		Non Isochronous: This value is in terms of number of descriptors. The values can be from 0 to 63. 0 - 1 descriptor. 63 - 64 descriptors. This field indicates the current descriptor processed in the list. This field is updated both by application and the core. For example, if the application enables the channel after programming CTD=5, then the core will start processing the 6th descriptor. The address is obtained by adding a value of (8bytes*5=) 40(decimal) to DMAAddr. Isochronous: CTD for isochronous is based on the current frame/(micro)frame value. Need to be set to zero by application.		

hcdmab1

Host Channel 1 DMA Buffer Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB0053C
i_usbotg_1_hostgrp	0xFFB40400	0xFFB4053C

Offset: 0x13C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcdmab1 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
hcdmab1 RW 0x0															

hcdmab1 Fields

Bit	Name	Description	Access	Reset
31:0	hcdmab1	Holds the current buffer address. This register is updated as and when the data transfer for the corresponding end point is in progress. This register is present only in Scatter/Gather DMA mode. Otherwise this field is reserved.	RW	0x0

hcchar2

Host Channel 2 Characteristics Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB00540
i_usbotg_1_hostgrp	0xFFB40400	0xFFB40540

Offset: 0x140

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
chena RW 0x0	chdis RW 0x0	oddfrm RW 0x0	devaddr RW 0x0							ec RW 0x0		eptype RW 0x0		lspddev RW 0x0	Reserved
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
epdir RW 0x0	epnum RW 0x0				mps RW 0x0										

hcchar2 Fields

Bit	Name	Description	Access	Reset
31	chena	Channel Enable (ChEna) When Scatter/Gather mode is enabled 1'b0: Indicates that the descriptor structure is not yet ready. 1'b1: Indicates that the descriptor structure and data buffer with data is setup and this channel can access the descriptor. When Scatter/Gather mode is disabled This field is set by the application and cleared by the OTG host. 1'b0: Channel disabled 1'b1: Channel enabled **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
30	chdis	Channel Disable (ChDis) The application sets this bit to stop transmitting/receiving data on a channel, even before the transfer For that channel is complete. The application must wait For the Channel Disabled interrupt before treating the channel as disabled. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
29	oddfrm	Odd Frame (OddFrm) This field is set (reset) by the application to indicate that the OTG host must perform a transfer in an odd (micro)frame. This field is applicable for only periodic (isochronous and interrupt) transactions. 1'b0: Even (micro)frame 1'b1: Odd (micro)frame	RW	0x0
28:22	devaddr	Device Address (DevAddr) This field selects the specific device serving as the data source or sink.	RW	0x0

Bit	Name	Description	Access	Reset
21:20	ec	Multi Count (MC) / Error Count (EC) When the Split Enable bit of the Host Channel-n Split Control register (HCSPLTn.SpltEna) is reset (1'b0), this field indicates to the host the number of transactions that must be executed per microframe For this periodic endpoint. For non periodic transfers, this field is used only in DMA mode, and specifies the number packets to be fetched For this channel before the internal DMA engine changes arbitration. 2'b00: Reserved This field yields undefined results. 2'b01: 1 transaction 2'b10: 2 transactions to be issued For this endpoint per microframe 2'b11: 3 transactions to be issued For this endpoint per microframe When HCSPLTn.SpltEna is Set (1'b1), this field indicates the number of immediate retries to be performed For a periodic split transactions on transaction errors. This field must be Set to at least 2'b01. **Value** **Description** 0x0 RESERVED 0x1 TRANSONE 0x2 TRANSTWO 0x3 TRANSTHREE	RW	0x0

Bit	Name	Description	Access	Reset
19:18	eptype	Endpoint Type (EPType) Indicates the transfer type selected. 2'b00: Control 2'b01: Isochronous 2'b10: Bulk 2'b11: Interrupt **Value** — **Description** 0x0 — CTRL 0x1 — ISOC 0x2 — BULK 0x3 — INTERR	RW	0x0
17	lspddev	Low-Speed Device (LSpdDev) This field is Set by the application to indicate that this channel is communicating to a low-speed device. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
15	epdir	Endpoint Direction (EPDir) Indicates whether the transaction is IN or OUT. 1'b0: OUT 1'b1: IN **Value** — **Description** 0x0 — OUT 0x1 — IN	RW	0x0

Bit	Name	Description	Access	Reset
14:11	epnum	Endpoint Number (EPNum) Indicates the endpoint number on the device serving as the data source or sink. Value — Description 0xa — ENDPT10 0xb — ENDPT11 0xc — ENDPT12 0xd — ENDPT13 0xe — ENDPT14 0xf — ENDPT15 0x0 — ENDPT0 0x1 — ENDPT1 0x2 — ENDPT2 0x3 — ENDPT3 0x4 — ENDPT4 0x5 — ENDPT5 0x6 — ENDPT6 0x7 — ENDPT7 0x8 — ENDPT8 0x9 — ENDPT9	RW	0x0
10:0	mps	Maximum Packet Size (MPS) Indicates the maximum packet size of the associated endpoint.	RW	0x0

hcsplt2

Host Channel 2 Split Control Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB00544
i_usbotg_1_hostgrp	0xFFB40400	0xFFB40544

Offset: 0x144

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
spltena RW 0x0	Reserved														compsplt RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
xactpos RW 0x0	hubaddr RW 0x0								prtaddr RW 0x0						

hcsplt2 Fields

Bit	Name	Description	Access	Reset
31	spltena	Split Enable (SpltEna) The application sets this field to indicate that this channel is enabled to perform split transactions. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
16	compsplt	Do Complete Split (CompSplt) The application sets this field to request the OTG host to perform a complete split transaction. **Value** — **Description** 0x0 — NOSPLIT 0x1 — SPLIT	RW	0x0

Bit	Name	Description	Access	Reset
15:14	xactpos	Transaction Position (XactPos) This field is used to determine whether to send all, first, middle, or last payloads with each OUT transaction. 2'b11: All. This is the entire data payload is of this transaction (which is less than or equal to 188 bytes). 2'b10: Begin. This is the first data payload of this transaction (which is larger than 188 bytes). 2'b00: Mid. This is the middle payload of this transaction (which is larger than 188 bytes). 2'b01: End. This is the last payload of this transaction (which is larger than 188 bytes). **Value** **Description** 0x0 MIDDLE 0x1 END 0x2 BEGIN 0x3 ALL	RW	0x0
13:7	hubaddr	Hub Address (HubAddr) This field holds the device address of the transaction translator's hub.	RW	0x0
6:0	prtaddr	Port Address (PrtAddr) This field is the port number of the recipient transaction translator.	RW	0x0

hcint2

Host Channel 2 Interrupt Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB00548

Module Instance	Base Address	Register Address
i_usbotg_1_hostgrp	0xFFB40400	0xFFB40548

Offset: 0x148

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		desc_lst_rollintr RW 0x0	xcs_xact_err RW 0x0	bnaintr RW 0x0	datatglerr RW 0x0	frmovrun RW 0x0	bblerr RW 0x0	xacterr RW 0x0	nyet RW 0x0	ack RW 0x0	nak RW 0x0	stall RW 0x0	ahberr RW 0x0	chhltd RW 0x0	xfercompl RW 0x0

hcint2 Fields

Bit	Name	Description	Access	Reset
13	desc_lst_rollintr	Descriptor rollover interrupt (DESC_LST_ROLLIntr) This bit is valid only when Scatter/Gather DMA mode is enabled. The core sets this bit when the corresponding channel's descriptor list rolls over. For non Scatter/Gather DMA mode, this bit is reserved. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
12	xcs_xact_err	Excessive Transaction Error (XCS_XACT_ERR) This bit is valid only when Scatter/Gather DMA mode is enabled. The core sets this bit when 3 consecutive transaction errors occurred on the USB bus. XCS_XACT_ERR will not be generated for Isochronous channels. For non Scatter/Gather DMA mode, this bit is reserved. **Value** **Description** 0x0 INACTIVE 0x1 ACVTIVE	RW	0x0
11	bnaintr	BNA (Buffer Not Available) Interrupt (BNAIntr) This bit is valid only when Scatter/Gather DMA mode is enabled. The core generates this interrupt when the descriptor accessed is not ready for the Core to process. BNA will not be generated for Isochronous channels. For non Scatter/Gather DMA mode, this bit is reserved. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
10	datatglerr	Data Toggle Error (DataTglErr).This bit can be set only by the core and the application should write 1 to clear it.In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
9	frmovrun	Frame Overrun (FrmOvrun).In Scatter/ Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the applica- tion should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
8	bblerr	Babble Error (BblErr) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core..This bit can be set only by the core and the applica- tion should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
7	xacterr	Transaction Error (XactErr) Indicates one of the following errors occurred on the USB. CRC check failure Timeout Bit stuff error False EOP In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the applica- tion should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
6	nyet	NYET Response Received Interrupt (NYET) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
5	ack	ACK Response Received/Transmitted Interrupt (ACK) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
4	nak	NAK Response Received Interrupt (NAK) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
3	stall	STALL Response Received Interrupt (STALL) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
2	ahberr	AHB Error (AHBErr) This is generated only in Internal DMA mode when there is an AHB error during AHB read/write. The application can read the corresponding channel's DMA address register to get the error address. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
1	chhltd	Channel Halted (ChHltd) In non Scatter/Gather DMA mode, it indicates the transfer completed abnormally either because of any USB transaction error or in response to disable request by the application or because of a completed transfer. in Scatter/gather DMA mode, this indicates that transfer completed due to any of the following . EOL being set in descriptor . AHB error . Excessive transaction errors . Babble . Stall **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
0	xfercompl	Transfer Completed (XferCompl) Transfer completed normally without any errors.This bit can be set only by the core and the application should write 1 to clear it. For Scatter/Gather DMA mode, it indicates that current descriptor processing got completed with IOC bit set in its descriptor. In non Scatter/Gather DMA mode, it indicates that Transfer completed normally without any errors. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

hcintmsk2

Host Channel 2 Interrupt Mask Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB0054C
i_usbotg_1_hostgrp	0xFFB40400	0xFFB4054C

Offset: 0x14C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		frm_lst_rollintrmsk RW 0x0	Reserved	bnaintrmsk RW 0x0	datatglerrmsk RW 0x0	frmovrunmsk RW 0x0	bblerrmsk RW 0x0	xacterrmsk RW 0x0	nyetmsk RW 0x0	ackmsk RW 0x0	nakmsk RW 0x0	stallmsk RW 0x0	ahberrmsk RW 0x0	chhltdmsk RW 0x0	xfercomplmsk RW 0x0

hcintmsk2 Fields

Bit	Name	Description	Access	Reset
13	frm_lst_rollintrmsk	Framelist rollover interrupt Mask register(FRM_LST_ROLLIntrMsk) This bit is valid only when Scatter/Gather DMA mode is enabled. Value: 0x0 — MASK; 0x1 — NOMASK	RW	0x0

Bit	Name	Description	Access	Reset
11	bnaintrmsk	BNA (Buffer Not Available) Interrupt mask register (BNAIntrMsk) This bit is valid only when Scatter/Gather DMA mode is enabled. Value: 0x0 Description: MASK Value: 0x1 Description: NOMASK	RW	0x0
10	datatglerrmsk	Data Toggle Error Mask (DataTglErrMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
9	frmovrunmsk	Frame Overrun Mask (FrmOvrunMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
8	bblerrmsk	Babble Error Mask (BblErrMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
7	xacterrmsk	Transaction Error Mask (XactErrMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0

Bit	Name	Description	Access	Reset
6	nyetmsk	NYET Response Received Interrupt Mask (NyetMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
5	ackmsk	ACK Response Received/Transmitted Interrupt Mask (AckMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
4	nakmsk	NAK Response Received Interrupt Mask (NakMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
3	stallmsk	STALL Response Received Interrupt Mask (StallMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0

Bit	Name	Description	Access	Reset
2	ahberrmsk	AHB Error Mask (AHBErrMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn. **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0
1	chhltdmsk	Channel Halted Mask (ChHltdMsk) **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0
0	xfercomplmsk	Transfer Completed Mask (XferComplMsk) **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0

hctsiz2

Host Channel 2 Transfer Size Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB00550
i_usbotg_1_hostgrp	0xFFB40400	0xFFB40550

Offset: 0x150

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dopng RW 0x0	pid RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
xfersize RW 0x0															

hctsiz2 Fields

Bit	Name	Description	Access	Reset
31	dopng	Do Ping (DoPng) This bit is used only For OUT transfers. Setting this field to 1 directs the host to do PING protocol. Note: Do not Set this bit For IN transfers. If this bit is Set For for IN transfers it disables the channel. **Value** — **Description** 0x0 — NOPING 0x1 — PING	RW	0x0

Bit	Name	Description	Access	Reset
30:29	pid	PID (Pid) The application programs this field with the type of PID to use For the initial transaction. The host maintains this field For the rest of the transfer. 2'b00: DATA0 2'b01: DATA2 2'b10: DATA1 2'b11: MDATA (non-control)/SETUP (control) **Value** **Description** 0x0 DATA0 0x1 DATA2 0x2 DATA1 0x3 MDATA	RW	0x0
28:19	pktcnt	Packet Count (PktCnt) This field is programmed by the application with the expected number of packets to be transmitted (OUT) or received (IN). The host decrements this count on every successful transmission or reception of an OUT/IN packet. Once this count reaches zero, the application is interrupted to indicate normal completion. The width of this counter is specified as Width of Packet Counters	RW	0x0
18:0	xfersize	Transfer Size (XferSize) For an OUT, this field is the number of data bytes the host sends during the transfer. For an IN, this field is the buffer size that the application has Reserved For the transfer. The application is expected to program this field as an integer multiple of the maximum packet size For IN transactions (periodic and non-periodic). The width of this counter is specified as Width of Transfer Size Counters	RW	0x0

hcdma2

Host Channel 2 DMA Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB00554
i_usbotg_1_hostgrp	0xFFB40400	0xFFB40554

Offset: 0x154

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcdma2 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
hcdma2 RW 0x0															

hcdma2 Fields

Bit	Name	Description	Access	Reset
31:0	hcdma2	Buffer DMA Mode: [31:0] DMA Address (DMAAddr) This field holds the start address in the external memory from which the data for the endpoint must be fetched or to which it must be stored. This register is incremented on every AHB transac- tion. Scatter-Gather DMA (DescDMA) Mode: [31:9] (Non Isoc) Non-Isochronous: [31:N] (Isoc) Isochronous: This field holds the start address of the 512 bytes page. The first descriptor in the list should be located in this address. The first	RW	0x0

Bit	Name	Description	Access	Reset
		descriptor may be or may not be ready. The core starts processing the list from the CTD value. This field holds the address of the 2*(nTD+1) bytes of locations in which the isochronous descriptors are present where N is based on nTD as per Table below [31:N] Base Address [N-1:3] Offset [2:0] 000 HS ISOC nTD N 7 6 15 7 31 8 63 9 127 10 255 11 FS ISOC nTD N 1 4 3 5 7 6 15 7 31 8 63 9 [N-1:3] (Isoc): [8:3] (Non Isoc): Current Transfer Desc(CTD): Non Isochronous: This value is in terms of number of descriptors. The values can be from 0 to 63. 0 - 1 descriptor. 63 - 64 descriptors. This field indicates the current descriptor processed in the list. This field is updated both by application and the core. For example, if the application enables the channel after programming CTD=5, then the core will start processing the 6th descriptor. The address is obtained		

Bit	Name	Description	Access	Reset
		by adding a value of (8bytes*5=) 40(decimal) to DMAAddr. Isochronous: CTD for isochronous is based on the current frame/(micro)frame value. Need to be set to zero by application.Scatter-Gather DMA (DescDMA) Mode: [31:9] (Non Isoc) Non-Isochronous: [31:N] (Isoc) Isochronous: This field holds the start address of the 512 bytes page. The first descriptor in the list should be located in this address. The first descriptor may be or may not be ready. The core starts processing the list from the CTD value. This field holds the address of the 2*(nTD+1) bytes of locations in which the isochronous descriptors are present where N is based on nTD as per Table below [31:N] Base Address [N-1:3] Offset [2:0] 000 HS ISOC nTD N 7 6 15 7 31 8 63 9 127 10 255 11 FS ISOC nTD N 1 4 3 5 7 6 15 7 31 8 63 9 [N-1:3] (Isoc): [8:3] (Non Isoc): Current Transfer Desc(CTD):		

Bit	Name	Description	Access	Reset
		Non Isochronous: This value is in terms of number of descriptors. The values can be from 0 to 63. 0 - 1 descriptor. 63 - 64 descriptors. This field indicates the current descriptor processed in the list. This field is updated both by application and the core. For example, if the application enables the channel after programming CTD=5, then the core will start processing the 6th descriptor. The address is obtained by adding a value of (8bytes*5=) 40(decimal) to DMAAddr. Isochronous: CTD for isochronous is based on the current frame/(micro)frame value. Need to be set to zero by application.		

hcdmab2

Host Channel 2 DMA Buffer Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB0055C
i_usbotg_1_hostgrp	0xFFB40400	0xFFB4055C

Offset: 0x15C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcdmab2 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
hcdmab2 RW 0x0															

hcdmab2 Fields

Bit	Name	Description	Access	Reset
31:0	hcdmab2	Holds the current buffer address. This register is updated as and when the data transfer for the corresponding end point is in progress. This register is present only in Scatter/Gather DMA mode. Otherwise this field is reserved.	RW	0x0

hcchar3

Host Channel 3 Characteristics Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB00560
i_usbotg_1_hostgrp	0xFFB40400	0xFFB40560

Offset: 0x160

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
chena RW 0x0	chdis RW 0x0	oddfrm RW 0x0	devaddr RW 0x0							ec RW 0x0		eptype RW 0x0		lspddev RW 0x0	Reserved
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
epdir RW 0x0	epnum RW 0x0				mps RW 0x0										

hcchar3 Fields

Bit	Name	Description	Access	Reset
31	chena	Channel Enable (ChEna) When Scatter/Gather mode is enabled 1'b0: Indicates that the descriptor structure is not yet ready. 1'b1: Indicates that the descriptor structure and data buffer with data is setup and this channel can access the descriptor. When Scatter/Gather mode is disabled This field is set by the application and cleared by the OTG host. 1'b0: Channel disabled 1'b1: Channel enabled **Value** **Description** 0x0 NOTREADY 0x1 READY	RW	0x0

Bit	Name	Description	Access	Reset
30	chdis	Channel Disable (ChDis) The application sets this bit to stop transmitting/receiving data on a channel, even before the transfer For that channel is complete. The application must wait For the Channel Disabled interrupt before treating the channel as disabled. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
29	oddfrm	Odd Frame (OddFrm) This field is set (reset) by the application to indicate that the OTG host must perform a transfer in an odd (micro)frame. This field is applicable for only periodic (isochronous and interrupt) transactions. 1'b0: Even (micro)frame 1'b1: Odd (micro)frame	RW	0x0
28:22	devaddr	Device Address (DevAddr) This field selects the specific device serving as the data source or sink.	RW	0x0

Bit	Name	Description	Access	Reset
21:20	ec	Multi Count (MC) / Error Count (EC) When the Split Enable bit of the Host Channel-n Split Control register (HCSPLTn.SpltEna) is reset (1'b0), this field indicates to the host the number of transactions that must be executed per microframe For this periodic endpoint. For non periodic transfers, this field is used only in DMA mode, and specifies the number packets to be fetched For this channel before the internal DMA engine changes arbitration. 2'b00: Reserved This field yields undefined results. 2'b01: 1 transaction 2'b10: 2 transactions to be issued For this endpoint per microframe 2'b11: 3 transactions to be issued For this endpoint per microframe When HCSPLTn.SpltEna is Set (1'b1), this field indicates the number of immediate retries to be performed For a periodic split transactions on transaction errors. This field must be Set to at least 2'b01. **Value** **Description** 0x0 RESERVED 0x1 TRANSONE 0x2 TRANSTWO 0x3 TRANSTHREE	RW	0x0

Bit	Name	Description	Access	Reset
19:18	eptype	Endpoint Type (EPType) Indicates the transfer type selected. 2'b00: Control 2'b01: Isochronous 2'b10: Bulk 2'b11: Interrupt **Value** / **Description** 0x0 — CTRL 0x1 — ISOC 0x2 — BULK 0x3 — INTERR	RW	0x0
17	lspddev	Low-Speed Device (LSpdDev) This field is Set by the application to indicate that this channel is communicating to a low-speed device. **Value** / **Description** 0x0 — NONLOWSPEED 0x1 — LOWSPEED	RW	0x0
15	epdir	Endpoint Direction (EPDir) Indicates whether the transaction is IN or OUT. 1'b0: OUT 1'b1: IN **Value** / **Description** 0x0 — OUTDIR 0x1 — INDIR	RW	0x0

Bit	Name	Description	Access	Reset
14:11	epnum	Endpoint Number (EPNum) Indicates the endpoint number on the device serving as the data source or sink. Value — Description 0xa — ENDPT10 0xb — ENDPT11 0xc — ENDPT12 0xd — ENDPT13 0xe — ENDPT14 0xf — ENDPT15 0x0 — ENDPT0 0x1 — ENDPT1 0x2 — ENDPT2 0x3 — ENDPT3 0x4 — ENDPT4 0x5 — ENDPT5 0x6 — ENDPT6 0x7 — ENDPT7 0x8 — ENDPT8 0x9 — ENDPT9	RW	0x0
10:0	mps	Maximum Packet Size (MPS) Indicates the maximum packet size of the associated endpoint.	RW	0x0

hcsplt3

Host Channel 3 Split Control Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB00564
i_usbotg_1_hostgrp	0xFFB40400	0xFFB40564

Offset: 0x164

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
spltena RW 0x0	Reserved														compsplt RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
xactpos RW 0x0	hubaddr RW 0x0								prtaddr RW 0x0						

hcsplt3 Fields

Bit	Name	Description	Access	Reset
31	spltena	Split Enable (SpltEna) The application sets this field to indicate that this channel is enabled to perform split transactions. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
16	compsplt	Do Complete Split (CompSplt) The application sets this field to request the OTG host to perform a complete split transaction. **Value** — **Description** 0x0 — NOSPLIT 0x1 — SPLIT	RW	0x0

Bit	Name	Description	Access	Reset
15:14	xactpos	Transaction Position (XactPos) This field is used to determine whether to send all, first, middle, or last payloads with each OUT transaction. 2'b11: All. This is the entire data payload is of this transaction (which is less than or equal to 188 bytes). 2'b10: Begin. This is the first data payload of this transaction (which is larger than 188 bytes). 2'b00: Mid. This is the middle payload of this transaction (which is larger than 188 bytes). 2'b01: End. This is the last payload of this transaction (which is larger than 188 bytes). **Value** **Description** 0x0 MIDDLE 0x1 END 0x2 BEGIN 0x3 ALL	RW	0x0
13:7	hubaddr	Hub Address (HubAddr) This field holds the device address of the transaction translator's hub.	RW	0x0
6:0	prtaddr	Port Address (PrtAddr) This field is the port number of the recipient transaction translator.	RW	0x0

hcint3

Host Channel 3 Interrupt Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB00568

Module Instance	Base Address	Register Address
i_usbotg_1_hostgrp	0xFFB40400	0xFFB40568

Offset: 0x168

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		desc_lst_rollintr RW 0x0	xcs_xact_err RW 0x0	bnaintr RW 0x0	datatglerr RW 0x0	frmovrun RW 0x0	bblerr RW 0x0	xacterr RW 0x0	nyet RW 0x0	ack RW 0x0	nak RW 0x0	stall RW 0x0	ahberr RW 0x0	chhltd RW 0x0	xfercompl RW 0x0

hcint3 Fields

Bit	Name	Description	Access	Reset
13	desc_lst_rollintr	Descriptor rollover interrupt (DESC_LST_ROLLIntr) This bit is valid only when Scatter/Gather DMA mode is enabled. The core sets this bit when the corresponding channel's descriptor list rolls over. For non Scatter/Gather DMA mode, this bit is reserved. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
12	xcs_xact_err	Excessive Transaction Error (XCS_XACT_ERR) This bit is valid only when Scatter/Gather DMA mode is enabled. The core sets this bit when 3 consecutive transaction errors occurred on the USB bus. XCS_XACT_ERR will not be generated for Isochronous channels. For non Scatter/Gather DMA mode, this bit is reserved. **Value** **Description** 0x0 INACTIVE 0x1 ACVTIVE	RW	0x0
11	bnaintr	BNA (Buffer Not Available) Interrupt (BNAIntr) This bit is valid only when Scatter/Gather DMA mode is enabled. The core generates this interrupt when the descriptor accessed is not ready for the Core to process. BNA will not be generated for Isochronous channels. For non Scatter/Gather DMA mode, this bit is reserved. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
10	datatglerr	Data Toggle Error (DataTglErr).This bit can be set only by the core and the application should write 1 to clear it.In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
9	frmovrun	Frame Overrun (FrmOvrun).In Scatter/ Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the applica- tion should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
8	bblerr	Babble Error (BblErr) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core..This bit can be set only by the core and the applica- tion should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
7	xacterr	Transaction Error (XactErr) Indicates one of the following errors occurred on the USB. CRC check failure Timeout Bit stuff error False EOP In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the applica- tion should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
6	nyet	NYET Response Received Interrupt (NYET) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
5	ack	ACK Response Received/Transmitted Interrupt (ACK) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
4	nak	NAK Response Received Interrupt (NAK) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
3	stall	STALL Response Received Interrupt (STALL) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
2	ahberr	AHB Error (AHBErr) This is generated only in Internal DMA mode when there is an AHB error during AHB read/write. The application can read the corresponding channel's DMA address register to get the error address. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
1	chhltd	Channel Halted (ChHltd) In non Scatter/Gather DMA mode, it indicates the transfer completed abnormally either because of any USB transaction error or in response to disable request by the application or because of a completed transfer. in Scatter/gather DMA mode, this indicates that transfer completed due to any of the following . EOL being set in descriptor . AHB error . Excessive transaction errors . Babble . Stall **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
0	xfercompl	Transfer Completed (XferCompl) Transfer completed normally without any errors.This bit can be set only by the core and the application should write 1 to clear it. For Scatter/Gather DMA mode, it indicates that current descriptor processing got completed with IOC bit set in its descriptor. In non Scatter/Gather DMA mode, it indicates that Transfer completed normally without any errors. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

hcintmsk3

Host Channel 3 Interrupt Mask Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB0056C
i_usbotg_1_hostgrp	0xFFB40400	0xFFB4056C

Offset: 0x16C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		frm_lst_rollintrmsk RW 0x0	Reserved	bnaintrmsk RW 0x0	datatglerrmsk RW 0x0	frmovrunmsk RW 0x0	bblerrmsk RW 0x0	xacterrmsk RW 0x0	nyetmsk RW 0x0	ackmsk RW 0x0	nakmsk RW 0x0	stallmsk RW 0x0	ahberrmsk RW 0x0	chhltdmsk RW 0x0	xfercomplmsk RW 0x0

hcintmsk3 Fields

Bit	Name	Description	Access	Reset
13	frm_lst_rollintrmsk	Framelist rollover interrupt Mask register(FRM_LST_ROLLIntrMsk) This bit is valid only when Scatter/Gather DMA mode is enabled. **Value** — **Description** 0x0 — MASK 0x1 — NOMASK	RW	0x0

Bit	Name	Description	Access	Reset
11	bnaintrmsk	BNA (Buffer Not Available) Interrupt mask register (BNAIntrMsk) This bit is valid only when Scatter/Gather DMA mode is enabled. Value: 0x0 Description: MASK Value: 0x1 Description: NOMASK	RW	0x0
10	datatglerrmsk	Data Toggle Error Mask (DataTglErrMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
9	frmovrunmsk	Frame Overrun Mask (FrmOvrunMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
8	bblerrmsk	Babble Error Mask (BblErrMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
7	xacterrmsk	Transaction Error Mask (XactErrMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0

Bit	Name	Description	Access	Reset
6	nyetmsk	NYET Response Received Interrupt Mask (NyetMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
5	ackmsk	ACK Response Received/Transmitted Interrupt Mask (AckMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
4	nakmsk	NAK Response Received Interrupt Mask (NakMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
3	stallmsk	STALL Response Received Interrupt Mask (StallMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0

Bit	Name	Description	Access	Reset
2	ahberrmsk	AHB Error Mask (AHBErrMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn. **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0
1	chhltdmsk	Channel Halted Mask (ChHltdMsk) **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0
0	xfercomplmsk	Transfer Completed Mask (XferComplMsk) **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0

hctsiz3

Host Channel 3 Transfer Size Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB00570
i_usbotg_1_hostgrp	0xFFB40400	0xFFB40570

Offset: 0x170

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dopng RW 0x0	pid RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
xfersize RW 0x0															

hctsiz3 Fields

Bit	Name	Description	Access	Reset
31	dopng	Do Ping (DoPng) This bit is used only For OUT transfers. Setting this field to 1 directs the host to do PING protocol. Note: Do not Set this bit For IN transfers. If this bit is Set For for IN transfers it disables the channel. **Value** — **Description** 0x0 — NOPING 0x1 — PING	RW	0x0

Bit	Name	Description	Access	Reset
30:29	pid	PID (Pid) The application programs this field with the type of PID to use For the initial transaction. The host maintains this field For the rest of the transfer. 2'b00: DATA0 2'b01: DATA2 2'b10: DATA1 2'b11: MDATA (non-control)/SETUP (control) **Value** **Description** 0x0 DATA0 0x1 DATA2 0x2 DATA1 0x3 MDATA	RW	0x0
28:19	pktcnt	Packet Count (PktCnt) This field is programmed by the application with the expected number of packets to be transmitted (OUT) or received (IN). The host decrements this count on every successful transmission or reception of an OUT/IN packet. Once this count reaches zero, the application is interrupted to indicate normal completion. The width of this counter is specified as Width of Packet Counters	RW	0x0
18:0	xfersize	Transfer Size (XferSize) For an OUT, this field is the number of data bytes the host sends during the transfer. For an IN, this field is the buffer size that the application has Reserved For the transfer. The application is expected to program this field as an integer multiple of the maximum packet size For IN transactions (periodic and non-periodic). The width of this counter is specified as Width of Transfer Size Counters	RW	0x0

hcdma3

Host Channel 3 DMA Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB00574
i_usbotg_1_hostgrp	0xFFB40400	0xFFB40574

Offset: 0x174

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcdma3 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
hcdma3 RW 0x0															

hcdma3 Fields

Bit	Name	Description	Access	Reset
31:0	hcdma3	Buffer DMA Mode: [31:0] DMA Address (DMAAddr) This field holds the start address in the external memory from which the data for the endpoint must be fetched or to which it must be stored. This register is incremented on every AHB transac- tion. Scatter-Gather DMA (DescDMA) Mode: [31:9] (Non Isoc) Non-Isochronous: [31:N] (Isoc) Isochronous: This field holds the start address of the 512 bytes page. The first descriptor in the list should be located in this address. The first	RW	0x0

Bit	Name	Description	Access	Reset
		descriptor may be or may not be ready. The core starts processing the list from the CTD value. This field holds the address of the 2*(nTD+1) bytes of locations in which the isochronous descriptors are present where N is based on nTD as per Table below [31:N] Base Address [N-1:3] Offset [2:0] 000 HS ISOC nTD N 7 6 15 7 31 8 63 9 127 10 255 11 FS ISOC nTD N 1 4 3 5 7 6 15 7 31 8 63 9 [N-1:3] (Isoc): [8:3] (Non Isoc): Current Transfer Desc(CTD): Non Isochronous: This value is in terms of number of descriptors. The values can be from 0 to 63. 0 - 1 descriptor. 63 - 64 descriptors. This field indicates the current descriptor processed in the list. This field is updated both by application and the core. For example, if the application enables the channel after programming CTD=5, then the core will start processing the 6th descriptor. The address is obtained		

Bit	Name	Description	Access	Reset
		by adding a value of (8bytes*5=) 40(decimal) to DMAAddr. Isochronous: CTD for isochronous is based on the current frame/(micro)frame value. Need to be set to zero by application.Scatter-Gather DMA (DescDMA) Mode: [31:9] (Non Isoc) Non-Isochronous: [31:N] (Isoc) Isochronous: This field holds the start address of the 512 bytes page. The first descriptor in the list should be located in this address. The first descriptor may be or may not be ready. The core starts processing the list from the CTD value. This field holds the address of the 2*(nTD+1) bytes of locations in which the isochronous descriptors are present where N is based on nTD as per Table below [31:N] Base Address [N-1:3] Offset [2:0] 000 HS ISOC nTD N 7 6 15 7 31 8 63 9 127 10 255 11 FS ISOC nTD N 1 4 3 5 7 6 15 7 31 8 63 9 [N-1:3] (Isoc): [8:3] (Non Isoc): Current Transfer Desc(CTD):		

Bit	Name	Description	Access	Reset
		Non Isochronous: This value is in terms of number of descriptors. The values can be from 0 to 63. 0 - 1 descriptor. 63 - 64 descriptors. This field indicates the current descriptor processed in the list. This field is updated both by application and the core. For example, if the application enables the channel after programming CTD=5, then the core will start processing the 6th descriptor. The address is obtained by adding a value of (8bytes*5=) 40(decimal) to DMAAddr. Isochronous: CTD for isochronous is based on the current frame/(micro)frame value. Need to be set to zero by application.		

hcdmab3

Host Channel 3 DMA Buffer Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB0057C
i_usbotg_1_hostgrp	0xFFB40400	0xFFB4057C

Offset: 0x17C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcdmab3 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
hcdmab3 RW 0x0															

hcdmab3 Fields

Bit	Name	Description	Access	Reset
31:0	hcdmab3	Holds the current buffer address. This register is updated as and when the data transfer for the corresponding end point is in progress. This register is present only in Scatter/Gather DMA mode. Otherwise this field is reserved.	RW	0x0

hcchar4

Host Channel 4 Characteristics Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB00580
i_usbotg_1_hostgrp	0xFFB40400	0xFFB40580

Offset: 0x180

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcdmab4 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
hcdmab4 RW 0x0															

hcchar4 Fields

Bit	Name	Description	Access	Reset
31:0	hcdmab4	These registers are present only in case of Scatter/Gather DMA. These registers are implemented in RAM instead of flop-based implementation. Holds the current buffer address. This register is updated as and when the data transfer for the corresponding end point is in progress. This register is present only in Scatter/Gather DMA mode. Otherwise this field is reserved.	RW	0x0

hcsplt4

Host Channel 4 Split Control Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB00584
i_usbotg_1_hostgrp	0xFFB40400	0xFFB40584

Offset: 0x184

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
spltena RW 0x0	Reserved														compsplt RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
xactpos RW 0x0		hubaddr RW 0x0							prtaddr RW 0x0						

hcsplt4 Fields

Bit	Name	Description	Access	Reset
31	spltena	Split Enable (SpltEna) The application sets this field to indicate that this channel is enabled to perform split transactions. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
16	compsplt	Do Complete Split (CompSplt) The application sets this field to request the OTG host to perform a complete split transaction. **Value** — **Description** 0x0 — NOSPLIT 0x1 — SPLIT	RW	0x0

Bit	Name	Description	Access	Reset
15:14	xactpos	Transaction Position (XactPos) This field is used to determine whether to send all, first, middle, or last payloads with each OUT transaction. 2'b11: All. This is the entire data payload is of this transaction (which is less than or equal to 188 bytes). 2'b10: Begin. This is the first data payload of this transaction (which is larger than 188 bytes). 2'b00: Mid. This is the middle payload of this transaction (which is larger than 188 bytes). 2'b01: End. This is the last payload of this transaction (which is larger than 188 bytes). **Value** **Description** 0x0 MIDDLE 0x1 END 0x2 BEGIN 0x3 ALL	RW	0x0
13:7	hubaddr	Hub Address (HubAddr) This field holds the device address of the transaction translator's hub.	RW	0x0
6:0	prtaddr	Port Address (PrtAddr) This field is the port number of the recipient transaction translator.	RW	0x0

hcint4

Host Channel 4 Interrupt Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB00588

Module Instance	Base Address	Register Address
i_usbotg_1_hostgrp	0xFFB40400	0xFFB40588

Offset: 0x188

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		desc_lst_rollintr RW 0x0	xcs_xact_err RW 0x0	bnaintr RW 0x0	datatglerr RW 0x0	frmovrun RW 0x0	bblerr RW 0x0	xacterr RW 0x0	nyet RW 0x0	ack RW 0x0	nak RW 0x0	stall RW 0x0	ahberr RW 0x0	chhltd RW 0x0	xfercompl RW 0x0

hcint4 Fields

Bit	Name	Description	Access	Reset
13	desc_lst_rollintr	Descriptor rollover interrupt (DESC_LST_ROLLIntr) This bit is valid only when Scatter/Gather DMA mode is enabled. The core sets this bit when the corresponding channel's descriptor list rolls over. For non Scatter/Gather DMA mode, this bit is reserved. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
12	xcs_xact_err	Excessive Transaction Error (XCS_XACT_ERR) This bit is valid only when Scatter/Gather DMA mode is enabled. The core sets this bit when 3 consecutive transaction errors occurred on the USB bus. XCS_XACT_ERR will not be generated for Isochronous channels. For non Scatter/Gather DMA mode, this bit is reserved. **Value** **Description** 0x0 INACTIVE 0x1 ACVTIVE	RW	0x0
11	bnaintr	BNA (Buffer Not Available) Interrupt (BNAIntr) This bit is valid only when Scatter/Gather DMA mode is enabled. The core generates this interrupt when the descriptor accessed is not ready for the Core to process. BNA will not be generated for Isochronous channels. For non Scatter/Gather DMA mode, this bit is reserved. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
10	datatglerr	Data Toggle Error (DataTglErr).This bit can be set only by the core and the application should write 1 to clear it.In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
9	frmovrun	Frame Overrun (FrmOvrun).In Scatter/ Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the applica- tion should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
8	bblerr	Babble Error (BblErr) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core..This bit can be set only by the core and the applica- tion should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
7	xacterr	Transaction Error (XactErr) Indicates one of the following errors occurred on the USB. CRC check failure Timeout Bit stuff error False EOP In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the applica- tion should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
6	nyet	NYET Response Received Interrupt (NYET) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
5	ack	ACK Response Received/Transmitted Interrupt (ACK) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
4	nak	NAK Response Received Interrupt (NAK) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
3	stall	STALL Response Received Interrupt (STALL) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
2	ahberr	AHB Error (AHBErr) This is generated only in Internal DMA mode when there is an AHB error during AHB read/write. The application can read the corresponding channel's DMA address register to get the error address. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
1	chhltd	Channel Halted (ChHltd) In non Scatter/Gather DMA mode, it indicates the transfer completed abnormally either because of any USB transaction error or in response to disable request by the application or because of a completed transfer. in Scatter/gather DMA mode, this indicates that transfer completed due to any of the following . EOL being set in descriptor . AHB error . Excessive transaction errors . Babble . Stall **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
0	xfercompl	Transfer Completed (XferCompl) Transfer completed normally without any errors.This bit can be set only by the core and the application should write 1 to clear it. For Scatter/Gather DMA mode, it indicates that current descriptor processing got completed with IOC bit set in its descriptor. In non Scatter/Gather DMA mode, it indicates that Transfer completed normally without any errors. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

hcintmsk4

Host Channel 4 Interrupt Mask Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB0058C
i_usbotg_1_hostgrp	0xFFB40400	0xFFB4058C

Offset: 0x18C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		frm_lst_rollintrmsk RW 0x0	Reserved	bnaintrmsk RW 0x0	datatglerrmsk RW 0x0	frmovrunmsk RW 0x0	bblerrmsk RW 0x0	xacterrmsk RW 0x0	nyetmsk RW 0x0	ackmsk RW 0x0	nakmsk RW 0x0	stallmsk RW 0x0	ahberrmsk RW 0x0	chhltdmsk RW 0x0	xfercomplmsk RW 0x0

hcintmsk4 Fields

Bit	Name	Description	Access	Reset
13	frm_lst_rollintrmsk	Framelist rollover interrupt Mask register(FRM_LST_ROLLIntrMsk) This bit is valid only when Scatter/Gather DMA mode is enabled. **Value** — **Description** 0x0 — MASK 0x1 — NOMASK	RW	0x0

Bit	Name	Description	Access	Reset
11	bnaintrmsk	BNA (Buffer Not Available) Interrupt mask register (BNAIntrMsk) This bit is valid only when Scatter/Gather DMA mode is enabled. Value: 0x0 Description: MASK Value: 0x1 Description: NOMASK	RW	0x0
10	datatglerrmsk	Data Toggle Error Mask (DataTglErrMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
9	frmovrunmsk	Frame Overrun Mask (FrmOvrunMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
8	bblerrmsk	Babble Error Mask (BblErrMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
7	xacterrmsk	Transaction Error Mask (XactErrMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0

Bit	Name	Description	Access	Reset
6	nyetmsk	NYET Response Received Interrupt Mask (NyetMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
5	ackmsk	ACK Response Received/Transmitted Interrupt Mask (AckMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
4	nakmsk	NAK Response Received Interrupt Mask (NakMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
3	stallmsk	STALL Response Received Interrupt Mask (StallMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0

Bit	Name	Description	Access	Reset
2	ahberrmsk	AHB Error Mask (AHBErrMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn. **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0
1	chhltdmsk	Channel Halted Mask (ChHltdMsk) **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0
0	xfercomplmsk	Transfer Completed Mask (XferComplMsk) **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0

hctsiz4

Host Channel 4 Transfer Size Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB00590
i_usbotg_1_hostgrp	0xFFB40400	0xFFB40590

Offset: 0x190

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dopng RW 0x0	pid RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
xfersize RW 0x0															

hctsiz4 Fields

Bit	Name	Description	Access	Reset
31	dopng	Do Ping (DoPng) This bit is used only For OUT transfers. Setting this field to 1 directs the host to do PING protocol. Note: Do not Set this bit For IN transfers. If this bit is Set For for IN transfers it disables the channel. **Value** — **Description** 0x0 — NOPING 0x1 — PING	RW	0x0

Bit	Name	Description	Access	Reset
30:29	pid	PID (Pid) The application programs this field with the type of PID to use For the initial transaction. The host maintains this field For the rest of the transfer. 2'b00: DATA0 2'b01: DATA2 2'b10: DATA1 2'b11: MDATA (non-control)/SETUP (control) **Value** **Description** 0x0 DATA0 0x1 DATA2 0x2 DATA1 0x3 MDATA	RW	0x0
28:19	pktcnt	Packet Count (PktCnt) This field is programmed by the application with the expected number of packets to be transmitted (OUT) or received (IN). The host decrements this count on every successful transmission or reception of an OUT/IN packet. Once this count reaches zero, the application is interrupted to indicate normal completion. The width of this counter is specified as Width of Packet Counters	RW	0x0
18:0	xfersize	Transfer Size (XferSize) For an OUT, this field is the number of data bytes the host sends during the transfer. For an IN, this field is the buffer size that the application has Reserved For the transfer. The application is expected to program this field as an integer multiple of the maximum packet size For IN transactions (periodic and non-periodic). The width of this counter is specified as Width of Transfer Size Counters	RW	0x0

hcdma4

Host Channel 4 DMA Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB00594
i_usbotg_1_hostgrp	0xFFB40400	0xFFB40594

Offset: 0x194

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcdma4 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
hcdma4 RW 0x0															

hcdma4 Fields

Bit	Name	Description	Access	Reset
31:0	hcdma4	Buffer DMA Mode: [31:0] DMA Address (DMAAddr) This field holds the start address in the external memory from which the data for the endpoint must be fetched or to which it must be stored. This register is incremented on every AHB transac- tion. Scatter-Gather DMA (DescDMA) Mode: [31:9] (Non Isoc) Non-Isochronous: [31:N] (Isoc) Isochronous: This field holds the start address of the 512 bytes page. The first descriptor in the list should be located in this address. The first	RW	0x0

Bit	Name	Description	Access	Reset
		descriptor may be or may not be ready. The core starts processing the list from the CTD value. This field holds the address of the 2*(nTD+1) bytes of locations in which the isochronous descriptors are present where N is based on nTD as per Table below [31:N] Base Address [N-1:3] Offset [2:0] 000 HS ISOC nTD N 7 6 15 7 31 8 63 9 127 10 255 11 FS ISOC nTD N 1 4 3 5 7 6 15 7 31 8 63 9 [N-1:3] (Isoc): [8:3] (Non Isoc): Current Transfer Desc(CTD): Non Isochronous: This value is in terms of number of descriptors. The values can be from 0 to 63. 0 - 1 descriptor. 63 - 64 descriptors. This field indicates the current descriptor processed in the list. This field is updated both by application and the core. For example, if the application enables the channel after programming CTD=5, then the core will start processing the 6th descriptor. The address is obtained		

Bit	Name	Description	Access	Reset
		by adding a value of (8bytes*5=) 40(decimal) to DMAAddr. Isochronous: CTD for isochronous is based on the current frame/(micro)frame value. Need to be set to zero by application.Scatter-Gather DMA (DescDMA) Mode: [31:9] (Non Isoc) Non-Isochronous: [31:N] (Isoc) Isochronous: This field holds the start address of the 512 bytes page. The first descriptor in the list should be located in this address. The first descriptor may be or may not be ready. The core starts processing the list from the CTD value. This field holds the address of the 2*(nTD+1) bytes of locations in which the isochronous descriptors are present where N is based on nTD as per Table below [31:N] Base Address [N-1:3] Offset [2:0] 000 HS ISOC nTD N 7 6 15 7 31 8 63 9 127 10 255 11 FS ISOC nTD N 1 4 3 5 7 6 15 7 31 8 63 9 [N-1:3] (Isoc): [8:3] (Non Isoc): Current Transfer Desc(CTD):		

Bit	Name	Description	Access	Reset
		Non Isochronous: This value is in terms of number of descriptors. The values can be from 0 to 63. 0 - 1 descriptor. 63 - 64 descriptors. This field indicates the current descriptor processed in the list. This field is updated both by application and the core. For example, if the application enables the channel after programming CTD=5, then the core will start processing the 6th descriptor. The address is obtained by adding a value of (8bytes*5=) 40(decimal) to DMAAddr. Isochronous: CTD for isochronous is based on the current frame/(micro)frame value. Need to be set to zero by application.		

hcdmab4

Host Channel 4 DMA Buffer Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB0059C
i_usbotg_1_hostgrp	0xFFB40400	0xFFB4059C

Offset: 0x19C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcdmab4 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
hcdmab4 RW 0x0															

hcdmab4 Fields

Bit	Name	Description	Access	Reset
31:0	hcdmab4	Holds the current buffer address. This register is updated as and when the data transfer for the corresponding end point is in progress. This register is present only in Scatter/Gather DMA mode. Otherwise this field is reserved.	RW	0x0

hcchar5

Host Channel 5 Characteristics Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB005A0
i_usbotg_1_hostgrp	0xFFB40400	0xFFB405A0

Offset: 0x1A0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
chena RW 0x0	chdis RW 0x0	oddfrm RW 0x0	devaddr RW 0x0							ec RW 0x0		eptype RW 0x0		lspddev RW 0x0	Reserved
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
epdir RW 0x0	epnum RW 0x0				mps RW 0x0										

hcchar5 Fields

Bit	Name	Description	Access	Reset
31	chena	Channel Enable (ChEna) When Scatter/Gather mode is enabled 1'b0: Indicates that the descriptor structure is not yet ready. 1'b1: Indicates that the descriptor structure and data buffer with data is setup and this channel can access the descriptor. When Scatter/Gather mode is disabled This field is set by the application and cleared by the OTG host. 1'b0: Channel disabled 1'b1: Channel enabled **Value** **Description** 0x0 NOTREADY 0x1 READY	RW	0x0

Bit	Name	Description	Access	Reset
30	chdis	Channel Disable (ChDis) The application sets this bit to stop transmitting/receiving data on a channel, even before the transfer For that channel is complete. The application must wait For the Channel Disabled interrupt before treating the channel as disabled. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
29	oddfrm	Odd Frame (OddFrm) This field is set (reset) by the application to indicate that the OTG host must perform a transfer in an odd (micro)frame. This field is applicable for only periodic (isochronous and interrupt) transactions. 1'b0: Even (micro)frame 1'b1: Odd (micro)frame	RW	0x0
28:22	devaddr	Device Address (DevAddr) This field selects the specific device serving as the data source or sink.	RW	0x0

Bit	Name	Description	Access	Reset
21:20	ec	Multi Count (MC) / Error Count (EC) When the Split Enable bit of the Host Channel-n Split Control register (HCSPLTn.SpltEna) is reset (1'b0), this field indicates to the host the number of transactions that must be executed per microframe For this periodic endpoint. For non periodic transfers, this field is used only in DMA mode, and specifies the number packets to be fetched For this channel before the internal DMA engine changes arbitration. 2'b00: Reserved This field yields undefined results. 2'b01: 1 transaction 2'b10: 2 transactions to be issued For this endpoint per microframe 2'b11: 3 transactions to be issued For this endpoint per microframe When HCSPLTn.SpltEna is Set (1'b1), this field indicates the number of immediate retries to be performed For a periodic split transactions on transaction errors. This field must be Set to at least 2'b01. **Value** **Description** 0x0 RESERVED 0x1 TRANSONE 0x2 TRANSTWO 0x3 TRANSTHREE	RW	0x0

Bit	Name	Description	Access	Reset
19:18	eptype	Endpoint Type (EPType) Indicates the transfer type selected. 2'b00: Control 2'b01: Isochronous 2'b10: Bulk 2'b11: Interrupt **Value** — **Description** 0x0 — CTRL 0x1 — ISOC 0x2 — BULK 0x3 — INTERR	RW	0x0
17	lspddev	Low-Speed Device (LSpdDev) This field is Set by the application to indicate that this channel is communicating to a low-speed device. **Value** — **Description** 0x0 — NONLOWSPEED 0x1 — LOWSPEED	RW	0x0
15	epdir	Endpoint Direction (EPDir) Indicates whether the transaction is IN or OUT. 1'b0: OUT 1'b1: IN **Value** — **Description** 0x0 — OUTDIR 0x1 — INDIR	RW	0x0

Bit	Name	Description	Access	Reset
14:11	epnum	Endpoint Number (EPNum) Indicates the endpoint number on the device serving as the data source or sink. Value — Description 0xa — ENDPT10 0xb — ENDPT11 0xc — ENDPT12 0xd — ENDPT13 0xe — ENDPT14 0xf — ENDPT15 0x0 — ENDPT0 0x1 — ENDPT1 0x2 — ENDPT2 0x3 — ENDPT3 0x4 — ENDPT4 0x5 — ENDPT5 0x6 — ENDPT6 0x7 — ENDPT7 0x8 — ENDPT8 0x9 — ENDPT9	RW	0x0
10:0	mps	Maximum Packet Size (MPS) Indicates the maximum packet size of the associated endpoint.	RW	0x0

hcsplt5

Host Channel 5 Split Control Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB005A4
i_usbotg_1_hostgrp	0xFFB40400	0xFFB405A4

Offset: 0x1A4

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
spltena RW 0x0	Reserved														compsplt RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
xactpos RW 0x0	hubaddr RW 0x0								prtaddr RW 0x0						

hcsplt5 Fields

Bit	Name	Description	Access	Reset
31	spltena	Split Enable (SpltEna) The application sets this field to indicate that this channel is enabled to perform split transactions. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
16	compsplt	Do Complete Split (CompSplt) The application sets this field to request the OTG host to perform a complete split transaction. **Value** — **Description** 0x0 — NOSPLIT 0x1 — SPLIT	RW	0x0

Bit	Name	Description	Access	Reset
15:14	xactpos	Transaction Position (XactPos) This field is used to determine whether to send all, first, middle, or last payloads with each OUT transaction. 2'b11: All. This is the entire data payload is of this transaction (which is less than or equal to 188 bytes). 2'b10: Begin. This is the first data payload of this transaction (which is larger than 188 bytes). 2'b00: Mid. This is the middle payload of this transaction (which is larger than 188 bytes). 2'b01: End. This is the last payload of this transaction (which is larger than 188 bytes). **Value** **Description** 0x0 MIDDLE 0x1 END 0x2 BEGIN 0x3 ALL	RW	0x0
13:7	hubaddr	Hub Address (HubAddr) This field holds the device address of the transaction translator's hub.	RW	0x0
6:0	prtaddr	Port Address (PrtAddr) This field is the port number of the recipient transaction translator.	RW	0x0

hcint5

Host Channel 5 Interrupt Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB005A8

Module Instance	Base Address	Register Address
i_usbotg_1_hostgrp	0xFFB40400	0xFFB405A8

Offset: 0x1A8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		desc_lst_rollintr RW 0x0	xcs_xact_err RW 0x0	bnaintr RW 0x0	datatglerr RW 0x0	frmovrun RW 0x0	bblerr RW 0x0	xacterr RW 0x0	nyet RW 0x0	ack RW 0x0	nak RW 0x0	stall RW 0x0	ahberr RW 0x0	chhltd RW 0x0	xfercompl RW 0x0

hcint5 Fields

Bit	Name	Description	Access	Reset
13	desc_lst_rollintr	Descriptor rollover interrupt (DESC_LST_ROLLIntr) This bit is valid only when Scatter/Gather DMA mode is enabled. The core sets this bit when the corresponding channel's descriptor list rolls over. For non Scatter/Gather DMA mode, this bit is reserved. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
12	xcs_xact_err	Excessive Transaction Error (XCS_XACT_ERR) This bit is valid only when Scatter/Gather DMA mode is enabled. The core sets this bit when 3 consecutive transaction errors occurred on the USB bus. XCS_XACT_ERR will not be generated for Isochronous channels. For non Scatter/Gather DMA mode, this bit is reserved. **Value** **Description** 0x0 INACTIVE 0x1 ACVTIVE	RW	0x0
11	bnaintr	BNA (Buffer Not Available) Interrupt (BNAIntr) This bit is valid only when Scatter/Gather DMA mode is enabled. The core generates this interrupt when the descriptor accessed is not ready for the Core to process. BNA will not be generated for Isochronous channels. For non Scatter/Gather DMA mode, this bit is reserved. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
10	datatglerr	Data Toggle Error (DataTglErr).This bit can be set only by the core and the application should write 1 to clear it.In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
9	frmovrun	Frame Overrun (FrmOvrun).In Scatter/ Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the applica- tion should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
8	bblerr	Babble Error (BblErr) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core..This bit can be set only by the core and the applica- tion should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
7	xacterr	Transaction Error (XactErr) Indicates one of the following errors occurred on the USB. CRC check failure Timeout Bit stuff error False EOP In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the applica- tion should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
6	nyet	NYET Response Received Interrupt (NYET) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
5	ack	ACK Response Received/Transmitted Interrupt (ACK) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
4	nak	NAK Response Received Interrupt (NAK) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
3	stall	STALL Response Received Interrupt (STALL) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
2	ahberr	AHB Error (AHBErr) This is generated only in Internal DMA mode when there is an AHB error during AHB read/write. The application can read the corresponding channel's DMA address register to get the error address. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
1	chhltd	Channel Halted (ChHltd) In non Scatter/Gather DMA mode, it indicates the transfer completed abnormally either because of any USB transaction error or in response to disable request by the application or because of a completed transfer. in Scatter/gather DMA mode, this indicates that transfer completed due to any of the following . EOL being set in descriptor . AHB error . Excessive transaction errors . Babble . Stall **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
0	xfercompl	Transfer Completed (XferCompl) Transfer completed normally without any errors.This bit can be set only by the core and the application should write 1 to clear it. For Scatter/Gather DMA mode, it indicates that current descriptor processing got completed with IOC bit set in its descriptor. In non Scatter/Gather DMA mode, it indicates that Transfer completed normally without any errors. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

hcintmsk5

Host Channel 5 Interrupt Mask Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB005AC
i_usbotg_1_hostgrp	0xFFB40400	0xFFB405AC

Offset: 0x1AC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		frm_lst_rollintrmsk RW 0x0	Reserved	bnaintrmsk RW 0x0	datatglerrmsk RW 0x0	frmovrunmsk RW 0x0	bblerrmsk RW 0x0	xacterrmsk RW 0x0	nyetmsk RW 0x0	ackmsk RW 0x0	nakmsk RW 0x0	stallmsk RW 0x0	ahberrmsk RW 0x0	chhltdmsk RW 0x0	xfercomplmsk RW 0x0

hcintmsk5 Fields

Bit	Name	Description	Access	Reset
13	frm_lst_rollintrmsk	Framelist rollover interrupt Mask register(FRM_LST_ROLLIntrMsk) This bit is valid only when Scatter/Gather DMA mode is enabled. **Value** — **Description** 0x0 — MASK 0x1 — NOMASK	RW	0x0

Bit	Name	Description	Access	Reset
11	bnaintrmsk	BNA (Buffer Not Available) Interrupt mask register (BNAIntrMsk) This bit is valid only when Scatter/Gather DMA mode is enabled. Value: 0x0 Description: MASK Value: 0x1 Description: NOMASK	RW	0x0
10	datatglerrmsk	Data Toggle Error Mask (DataTglErrMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
9	frmovrunmsk	Frame Overrun Mask (FrmOvrunMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
8	bblerrmsk	Babble Error Mask (BblErrMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
7	xacterrmsk	Transaction Error Mask (XactErrMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0

Bit	Name	Description	Access	Reset
6	nyetmsk	NYET Response Received Interrupt Mask (NyetMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
5	ackmsk	ACK Response Received/Transmitted Interrupt Mask (AckMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
4	nakmsk	NAK Response Received Interrupt Mask (NakMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
3	stallmsk	STALL Response Received Interrupt Mask (StallMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0

Bit	Name	Description	Access	Reset
2	ahberrmsk	AHB Error Mask (AHBErrMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn. **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0
1	chhltdmsk	Channel Halted Mask (ChHltdMsk) **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0
0	xfercomplmsk	Transfer Completed Mask (XferComplMsk) **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0

hctsiz5

Host Channel 5 Transfer Size Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB005B0
i_usbotg_1_hostgrp	0xFFB40400	0xFFB405B0

Offset: 0x1B0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dopng RW 0x0	pid RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
xfersize RW 0x0															

hctsiz5 Fields

Bit	Name	Description	Access	Reset
31	dopng	Do Ping (DoPng) This bit is used only For OUT transfers. Setting this field to 1 directs the host to do PING protocol. Note: Do not Set this bit For IN transfers. If this bit is Set For for IN transfers it disables the channel. **Value** — **Description** 0x0 — NOPING 0x1 — PING	RW	0x0

Bit	Name	Description	Access	Reset
30:29	pid	PID (Pid) The application programs this field with the type of PID to use For the initial transaction. The host maintains this field For the rest of the transfer. 2'b00: DATA0 2'b01: DATA2 2'b10: DATA1 2'b11: MDATA (non-control)/SETUP (control) **Value** **Description** 0x0 DATA0 0x1 DATA2 0x2 DATA1 0x3 MDATA	RW	0x0
28:19	pktcnt	Packet Count (PktCnt) This field is programmed by the application with the expected number of packets to be transmitted (OUT) or received (IN). The host decrements this count on every successful transmission or reception of an OUT/IN packet. Once this count reaches zero, the application is interrupted to indicate normal completion. The width of this counter is specified as Width of Packet Counters	RW	0x0
18:0	xfersize	Transfer Size (XferSize) For an OUT, this field is the number of data bytes the host sends during the transfer. For an IN, this field is the buffer size that the application has Reserved For the transfer. The application is expected to program this field as an integer multiple of the maximum packet size For IN transactions (periodic and non-periodic). The width of this counter is specified as Width of Transfer Size Counters	RW	0x0

hcdma5

Host Channel 5 DMA Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB005B4
i_usbotg_1_hostgrp	0xFFB40400	0xFFB405B4

Offset: 0x1B4

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcdma5 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
hcdma5 RW 0x0															

hcdma5 Fields

Bit	Name	Description	Access	Reset
31:0	hcdma5	Buffer DMA Mode: [31:0] DMA Address (DMAAddr) This field holds the start address in the external memory from which the data for the endpoint must be fetched or to which it must be stored. This register is incremented on every AHB transac- tion. Scatter-Gather DMA (DescDMA) Mode: [31:9] (Non Isoc) Non-Isochronous: [31:N] (Isoc) Isochronous: This field holds the start address of the 512 bytes page. The first descriptor in the list should be located in this address. The first	RW	0x0

Bit	Name	Description	Access	Reset
		descriptor may be or may not be ready. The core starts processing the list from the CTD value. This field holds the address of the 2*(nTD+1) bytes of locations in which the isochronous descriptors are present where N is based on nTD as per Table below [31:N] Base Address [N-1:3] Offset [2:0] 000 HS ISOC nTD N 7 6 15 7 31 8 63 9 127 10 255 11 FS ISOC nTD N 1 4 3 5 7 6 15 7 31 8 63 9 [N-1:3] (Isoc): [8:3] (Non Isoc): Current Transfer Desc(CTD): Non Isochronous: This value is in terms of number of descriptors. The values can be from 0 to 63. 0 - 1 descriptor. 63 - 64 descriptors. This field indicates the current descriptor processed in the list. This field is updated both by application and the core. For example, if the application enables the channel after programming CTD=5, then the core will start processing the 6th descriptor. The address is obtained		

Bit	Name	Description	Access	Reset
		by adding a value of (8bytes*5=) 40(decimal) to DMAAddr. Isochronous: CTD for isochronous is based on the current frame/(micro)frame value. Need to be set to zero by application.Scatter-Gather DMA (DescDMA) Mode: [31:9] (Non Isoc) Non-Isochronous: [31:N] (Isoc) Isochronous: This field holds the start address of the 512 bytes page. The first descriptor in the list should be located in this address. The first descriptor may be or may not be ready. The core starts processing the list from the CTD value. This field holds the address of the 2*(nTD+1) bytes of locations in which the isochronous descriptors are present where N is based on nTD as per Table below [31:N] Base Address [N-1:3] Offset [2:0] 000 HS ISOC nTD N 7 6 15 7 31 8 63 9 127 10 255 11 FS ISOC nTD N 1 4 3 5 7 6 15 7 31 8 63 9 [N-1:3] (Isoc): [8:3] (Non Isoc): Current Transfer Desc(CTD):		

Bit	Name	Description	Access	Reset
		Non Isochronous: This value is in terms of number of descriptors. The values can be from 0 to 63. 0 - 1 descriptor. 63 - 64 descriptors. This field indicates the current descriptor processed in the list. This field is updated both by application and the core. For example, if the application enables the channel after programming CTD=5, then the core will start processing the 6th descriptor. The address is obtained by adding a value of (8bytes*5=) 40(decimal) to DMAAddr. Isochronous: CTD for isochronous is based on the current frame/(micro)frame value. Need to be set to zero by application.		

hcdmab5

Host Channel 5 DMA Buffer Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB005BC
i_usbotg_1_hostgrp	0xFFB40400	0xFFB405BC

Offset: 0x1BC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcdmab5 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
hcdmab5 RW 0x0															

hcdmab5 Fields

Bit	Name	Description	Access	Reset
31:0	hcdmab5	Holds the current buffer address. This register is updated as and when the data transfer for the corresponding end point is in progress. This register is present only in Scatter/Gather DMA mode. Otherwise this field is reserved.	RW	0x0

hcchar6

Host Channel 6 Characteristics Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB005C0
i_usbotg_1_hostgrp	0xFFB40400	0xFFB405C0

Offset: 0x1C0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
chena RW 0x0	chdis RW 0x0	oddfrm RW 0x0	devaddr RW 0x0							ec RW 0x0		eptype RW 0x0		lspddev RW 0x0	Reserved
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
epdir RW 0x0	epnum RW 0x0				mps RW 0x0										

hcchar6 Fields

Bit	Name	Description	Access	Reset
31	chena	Channel Enable (ChEna) When Scatter/Gather mode is enabled 1'b0: Indicates that the descriptor structure is not yet ready. 1'b1: Indicates that the descriptor structure and data buffer with data is setup and this channel can access the descriptor. When Scatter/Gather mode is disabled This field is set by the application and cleared by the OTG host. 1'b0: Channel disabled 1'b1: Channel enabled **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
30	chdis	Channel Disable (ChDis) The application sets this bit to stop transmitting/receiving data on a channel, even before the transfer For that channel is complete. The application must wait For the Channel Disabled interrupt before treating the channel as disabled. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
29	oddfrm	Odd Frame (OddFrm) This field is set (reset) by the application to indicate that the OTG host must perform a transfer in an odd (micro)frame. This field is applicable for only periodic (isochronous and interrupt) transactions. 1'b0: Even (micro)frame 1'b1: Odd (micro)frame	RW	0x0
28:22	devaddr	Device Address (DevAddr) This field selects the specific device serving as the data source or sink.	RW	0x0

Bit	Name	Description	Access	Reset
21:20	ec	Multi Count (MC) / Error Count (EC) When the Split Enable bit of the Host Channel-n Split Control register (HCSPLTn.SpltEna) is reset (1'b0), this field indicates to the host the number of transactions that must be executed per microframe For this periodic endpoint. For non periodic transfers, this field is used only in DMA mode, and specifies the number packets to be fetched For this channel before the internal DMA engine changes arbitration. 2'b00: Reserved This field yields undefined results. 2'b01: 1 transaction 2'b10: 2 transactions to be issued For this endpoint per microframe 2'b11: 3 transactions to be issued For this endpoint per microframe When HCSPLTn.SpltEna is Set (1'b1), this field indicates the number of immediate retries to be performed For a periodic split transactions on transaction errors. This field must be Set to at least 2'b01. **Value** **Description** 0x0 RESERVED 0x1 TRANSONE 0x2 TRANSTWO 0x3 TRANSTHREE	RW	0x0

Bit	Name	Description	Access	Reset
19:18	eptype	Endpoint Type (EPType) Indicates the transfer type selected. 2'b00: Control 2'b01: Isochronous 2'b10: Bulk 2'b11: Interrupt **Value** — **Description** 0x0 — CTRL 0x1 — ISOC 0x2 — BULK 0x3 — INTERR	RW	0x0
17	lspddev	Low-Speed Device (LSpdDev) This field is Set by the application to indicate that this channel is communicating to a low-speed device. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
15	epdir	Endpoint Direction (EPDir) Indicates whether the transaction is IN or OUT. 1'b0: OUT 1'b1: IN **Value** — **Description** 0x0 — OUT 0x1 — IN	RW	0x0

Bit	Name	Description	Access	Reset
14:11	epnum	Endpoint Number (EPNum) Indicates the endpoint number on the device serving as the data source or sink. Value — Description 0xa — ENDPT10 0xb — ENDPT11 0xc — ENDPT12 0xd — ENDPT13 0xe — ENDPT14 0xf — ENDPT15 0x0 — ENDPT0 0x1 — ENDPT1 0x2 — ENDPT2 0x3 — ENDPT3 0x4 — ENDPT4 0x5 — ENDPT5 0x6 — ENDPT6 0x7 — ENDPT7 0x8 — ENDPT8 0x9 — ENDPT9	RW	0x0
10:0	mps	Maximum Packet Size (MPS) Indicates the maximum packet size of the associated endpoint.	RW	0x0

hcsplt6

Host Channel 6 Split Control Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB005C4
i_usbotg_1_hostgrp	0xFFB40400	0xFFB405C4

Offset: 0x1C4

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
spltena RW 0x0	Reserved														compsplt RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
xactpos RW 0x0		hubaddr RW 0x0							prtaddr RW 0x0						

hcsplt6 Fields

Bit	Name	Description	Access	Reset
31	spltena	Split Enable (SpltEna) The application sets this field to indicate that this channel is enabled to perform split transactions. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
16	compsplt	Do Complete Split (CompSplt) The application sets this field to request the OTG host to perform a complete split transaction. **Value** — **Description** 0x0 — NOSPLIT 0x1 — SPLIT	RW	0x0

Bit	Name	Description	Access	Reset
15:14	xactpos	Transaction Position (XactPos) This field is used to determine whether to send all, first, middle, or last payloads with each OUT transaction. 2'b11: All. This is the entire data payload is of this transaction (which is less than or equal to 188 bytes). 2'b10: Begin. This is the first data payload of this transaction (which is larger than 188 bytes). 2'b00: Mid. This is the middle payload of this transaction (which is larger than 188 bytes). 2'b01: End. This is the last payload of this transaction (which is larger than 188 bytes). **Value** **Description** 0x0 MIDDLE 0x1 END 0x2 BEGIN 0x3 ALL	RW	0x0
13:7	hubaddr	Hub Address (HubAddr) This field holds the device address of the transaction translator's hub.	RW	0x0
6:0	prtaddr	Port Address (PrtAddr) This field is the port number of the recipient transaction translator.	RW	0x0

hcint6

Host Channel 6 Interrupt Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB005C8

Module Instance	Base Address	Register Address
i_usbotg_1_hostgrp	0xFFB40400	0xFFB405C8

Offset: 0x1C8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		desc_lst_rollintr RW 0x0	xcs_xact_err RW 0x0	bnaintr RW 0x0	datatglerr RW 0x0	frmovrun RW 0x0	bblerr RW 0x0	xacterr RW 0x0	nyet RW 0x0	ack RW 0x0	nak RW 0x0	stall RW 0x0	ahberr RW 0x0	chhltd RW 0x0	xfercompl RW 0x0

hcint6 Fields

Bit	Name	Description	Access	Reset
13	desc_lst_rollintr	Descriptor rollover interrupt (DESC_LST_ROLLIntr) This bit is valid only when Scatter/Gather DMA mode is enabled. The core sets this bit when the corresponding channel's descriptor list rolls over. For non Scatter/Gather DMA mode, this bit is reserved. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
12	xcs_xact_err	Excessive Transaction Error (XCS_XACT_ERR) This bit is valid only when Scatter/Gather DMA mode is enabled. The core sets this bit when 3 consecutive transaction errors occurred on the USB bus. XCS_XACT_ERR will not be generated for Isochronous channels. For non Scatter/Gather DMA mode, this bit is reserved. **Value** **Description** 0x0 INACTIVE 0x1 ACVTIVE	RW	0x0
11	bnaintr	BNA (Buffer Not Available) Interrupt (BNAIntr) This bit is valid only when Scatter/Gather DMA mode is enabled. The core generates this interrupt when the descriptor accessed is not ready for the Core to process. BNA will not be generated for Isochronous channels. For non Scatter/Gather DMA mode, this bit is reserved. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
10	datatglerr	Data Toggle Error (DataTglErr).This bit can be set only by the core and the application should write 1 to clear it.In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
9	frmovrun	Frame Overrun (FrmOvrun).In Scatter/ Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the applica- tion should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
8	bblerr	Babble Error (BblErr) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core..This bit can be set only by the core and the applica- tion should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
7	xacterr	Transaction Error (XactErr) Indicates one of the following errors occurred on the USB. CRC check failure Timeout Bit stuff error False EOP In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the applica- tion should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
6	nyet	NYET Response Received Interrupt (NYET) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
5	ack	ACK Response Received/Transmitted Interrupt (ACK) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
4	nak	NAK Response Received Interrupt (NAK) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
3	stall	STALL Response Received Interrupt (STALL) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
2	ahberr	AHB Error (AHBErr) This is generated only in Internal DMA mode when there is an AHB error during AHB read/write. The application can read the corresponding channel's DMA address register to get the error address. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
1	chhltd	Channel Halted (ChHltd) In non Scatter/Gather DMA mode, it indicates the transfer completed abnormally either because of any USB transaction error or in response to disable request by the application or because of a completed transfer. in Scatter/gather DMA mode, this indicates that transfer completed due to any of the following . EOL being set in descriptor . AHB error . Excessive transaction errors . Babble . Stall **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
0	xfercompl	Transfer Completed (XferCompl) Transfer completed normally without any errors.This bit can be set only by the core and the application should write 1 to clear it. For Scatter/Gather DMA mode, it indicates that current descriptor processing got completed with IOC bit set in its descriptor. In non Scatter/Gather DMA mode, it indicates that Transfer completed normally without any errors. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

hcintmsk6

Host Channel 6 Interrupt Mask Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB005CC
i_usbotg_1_hostgrp	0xFFB40400	0xFFB405CC

Offset: 0x1CC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		frm_lst_rollintrmsk RW 0x0	Reserved	bnaintrmsk RW 0x0	datatglerrmsk RW 0x0	frmovrunmsk RW 0x0	bblerrmsk RW 0x0	xacterrmsk RW 0x0	nyetmsk RW 0x0	ackmsk RW 0x0	nakmsk RW 0x0	stallmsk RW 0x0	ahberrmsk RW 0x0	chhltdmsk RW 0x0	xfercomplmsk RW 0x0

hcintmsk6 Fields

Bit	Name	Description	Access	Reset
13	frm_lst_rollintrmsk	Framelist rollover interrupt Mask register(FRM_LST_ROLLIntrMsk) This bit is valid only when Scatter/Gather DMA mode is enabled. **Value** — **Description** 0x0 — MASK 0x1 — NOMASK	RW	0x0

Bit	Name	Description	Access	Reset
11	bnaintrmsk	BNA (Buffer Not Available) Interrupt mask register (BNAIntrMsk) This bit is valid only when Scatter/Gather DMA mode is enabled. Value: 0x0 — MASK; 0x1 — NOMASK	RW	0x0
10	datatglerrmsk	Data Toggle Error Mask (DataTglErrMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
9	frmovrunmsk	Frame Overrun Mask (FrmOvrunMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
8	bblerrmsk	Babble Error Mask (BblErrMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
7	xacterrmsk	Transaction Error Mask (XactErrMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0

Bit	Name	Description	Access	Reset
6	nyetmsk	NYET Response Received Interrupt Mask (NyetMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
5	ackmsk	ACK Response Received/Transmitted Interrupt Mask (AckMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
4	nakmsk	NAK Response Received Interrupt Mask (NakMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
3	stallmsk	STALL Response Received Interrupt Mask (StallMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0

Bit	Name	Description	Access	Reset
2	ahberrmsk	AHB Error Mask (AHBErrMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn. **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0
1	chhltdmsk	Channel Halted Mask (ChHltdMsk) **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0
0	xfercomplmsk	Transfer Completed Mask (XferComplMsk) **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0

hctsiz6

Host Channel 6 Transfer Size Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB005D0
i_usbotg_1_hostgrp	0xFFB40400	0xFFB405D0

Offset: 0x1D0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dopng RW 0x0	pid RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
xfersize RW 0x0															

hctsiz6 Fields

Bit	Name	Description	Access	Reset
31	dopng	Do Ping (DoPng) This bit is used only For OUT transfers. Setting this field to 1 directs the host to do PING protocol. Note: Do not Set this bit For IN transfers. If this bit is Set For for IN transfers it disables the channel. **Value** — **Description** 0x0 — NOPING 0x1 — PING	RW	0x0

Bit	Name	Description	Access	Reset
30:29	pid	PID (Pid) The application programs this field with the type of PID to use For the initial transaction. The host maintains this field For the rest of the transfer. 2'b00: DATA0 2'b01: DATA2 2'b10: DATA1 2'b11: MDATA (non-control)/SETUP (control) **Value** **Description** 0x0 DATA0 0x1 DATA2 0x2 DATA1 0x3 MDATA	RW	0x0
28:19	pktcnt	Packet Count (PktCnt) This field is programmed by the application with the expected number of packets to be transmitted (OUT) or received (IN). The host decrements this count on every successful transmission or reception of an OUT/IN packet. Once this count reaches zero, the application is interrupted to indicate normal completion. The width of this counter is specified as Width of Packet Counters	RW	0x0
18:0	xfersize	Transfer Size (XferSize) For an OUT, this field is the number of data bytes the host sends during the transfer. For an IN, this field is the buffer size that the application has Reserved For the transfer. The application is expected to program this field as an integer multiple of the maximum packet size For IN transactions (periodic and non-periodic). The width of this counter is specified as Width of Transfer Size Counters	RW	0x0

hcdma6

Host Channel 6 DMA Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB005D4
i_usbotg_1_hostgrp	0xFFB40400	0xFFB405D4

Offset: 0x1D4

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcdma6 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
hcdma6 RW 0x0															

hcdma6 Fields

Bit	Name	Description	Access	Reset
31:0	hcdma6	Buffer DMA Mode: [31:0] DMA Address (DMAAddr) This field holds the start address in the external memory from which the data for the endpoint must be fetched or to which it must be stored. This register is incremented on every AHB transac- tion. Scatter-Gather DMA (DescDMA) Mode: [31:9] (Non Isoc) Non-Isochronous: [31:N] (Isoc) Isochronous: This field holds the start address of the 512 bytes page. The first descriptor in the list should be located in this address. The first	RW	0x0

Bit	Name	Description	Access	Reset
		descriptor may be or may not be ready. The core starts processing the list from the CTD value. This field holds the address of the 2*(nTD+1) bytes of locations in which the isochronous descriptors are present where N is based on nTD as per Table below [31:N] Base Address [N-1:3] Offset [2:0] 000 HS ISOC nTD N 7 6 15 7 31 8 63 9 127 10 255 11 FS ISOC nTD N 1 4 3 5 7 6 15 7 31 8 63 9 [N-1:3] (Isoc): [8:3] (Non Isoc): Current Transfer Desc(CTD): Non Isochronous: This value is in terms of number of descriptors. The values can be from 0 to 63. 0 - 1 descriptor. 63 - 64 descriptors. This field indicates the current descriptor processed in the list. This field is updated both by application and the core. For example, if the application enables the channel after programming CTD=5, then the core will start processing the 6th descriptor. The address is obtained		

Bit	Name	Description	Access	Reset
		by adding a value of (8bytes*5=) 40(decimal) to DMAAddr. Isochronous: CTD for isochronous is based on the current frame/(micro)frame value. Need to be set to zero by application.Scatter-Gather DMA (DescDMA) Mode: [31:9] (Non Isoc) Non-Isochronous: [31:N] (Isoc) Isochronous: This field holds the start address of the 512 bytes page. The first descriptor in the list should be located in this address. The first descriptor may be or may not be ready. The core starts processing the list from the CTD value. This field holds the address of the 2*(nTD+1) bytes of locations in which the isochronous descriptors are present where N is based on nTD as per Table below [31:N] Base Address [N-1:3] Offset [2:0] 000 HS ISOC nTD N 7 6 15 7 31 8 63 9 127 10 255 11 FS ISOC nTD N 1 4 3 5 7 6 15 7 31 8 63 9 [N-1:3] (Isoc): [8:3] (Non Isoc): Current Transfer Desc(CTD):		

Bit	Name	Description	Access	Reset
		Non Isochronous: This value is in terms of number of descriptors. The values can be from 0 to 63. 0 - 1 descriptor. 63 - 64 descriptors. This field indicates the current descriptor processed in the list. This field is updated both by application and the core. For example, if the application enables the channel after programming CTD=5, then the core will start processing the 6th descriptor. The address is obtained by adding a value of (8bytes*5=) 40(decimal) to DMAAddr. Isochronous: CTD for isochronous is based on the current frame/(micro)frame value. Need to be set to zero by application.		

hcdmab6

Host Channel 6 DMA Buffer Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB005DC
i_usbotg_1_hostgrp	0xFFB40400	0xFFB405DC

Offset: 0x1DC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcdmab6 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
hcdmab6 RW 0x0															

hcdmab6 Fields

Bit	Name	Description	Access	Reset
31:0	hcdmab6	Holds the current buffer address. This register is updated as and when the data transfer for the corresponding end point is in progress. This register is present only in Scatter/Gather DMA mode. Otherwise this field is reserved.	RW	0x0

hcchar7

Host Channel 7 Characteristics Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB005E0
i_usbotg_1_hostgrp	0xFFB40400	0xFFB405E0

Offset: 0x1E0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
chena RW 0x0	chdis RW 0x0	oddfrm RW 0x0	devaddr RW 0x0							ec RW 0x0		eptype RW 0x0		lspddev RW 0x0	Reserved
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
epdir RW 0x0	epnum RW 0x0				mps RW 0x0										

hcchar7 Fields

Bit	Name	Description	Access	Reset
31	chena	Channel Enable (ChEna) When Scatter/Gather mode is enabled 1'b0: Indicates that the descriptor structure is not yet ready. 1'b1: Indicates that the descriptor structure and data buffer with data is setup and this channel can access the descriptor. When Scatter/Gather mode is disabled This field is set by the application and cleared by the OTG host. 1'b0: Channel disabled 1'b1: Channel enabled **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
30	chdis	Channel Disable (ChDis) The application sets this bit to stop transmitting/receiving data on a channel, even before the transfer For that channel is complete. The application must wait For the Channel Disabled interrupt before treating the channel as disabled. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
29	oddfrm	Odd Frame (OddFrm) This field is set (reset) by the application to indicate that the OTG host must perform a transfer in an odd (micro)frame. This field is applicable for only periodic (isochronous and interrupt) transactions. 1'b0: Even (micro)frame 1'b1: Odd (micro)frame	RW	0x0
28:22	devaddr	Device Address (DevAddr) This field selects the specific device serving as the data source or sink.	RW	0x0

Bit	Name	Description	Access	Reset
21:20	ec	Multi Count (MC) / Error Count (EC) When the Split Enable bit of the Host Channel-n Split Control register (HCSPLTn.SpltEna) is reset (1'b0), this field indicates to the host the number of transactions that must be executed per microframe For this periodic endpoint. For non periodic transfers, this field is used only in DMA mode, and specifies the number packets to be fetched For this channel before the internal DMA engine changes arbitration. 2'b00: Reserved This field yields undefined results. 2'b01: 1 transaction 2'b10: 2 transactions to be issued For this endpoint per microframe 2'b11: 3 transactions to be issued For this endpoint per microframe When HCSPLTn.SpltEna is Set (1'b1), this field indicates the number of immediate retries to be performed For a periodic split transactions on transaction errors. This field must be Set to at least 2'b01. **Value** **Description** 0x0 RESERVED 0x1 TRANSONE 0x2 TRANSTWO 0x3 TRANSTHREE	RW	0x0

Bit	Name	Description	Access	Reset
19:18	eptype	Endpoint Type (EPType) Indicates the transfer type selected. 2'b00: Control 2'b01: Isochronous 2'b10: Bulk 2'b11: Interrupt **Value** **Description** 0x0 CTRL 0x1 ISOC 0x2 BULK 0x3 INTERR	RW	0x0
17	lspddev	Low-Speed Device (LSpdDev) This field is Set by the application to indicate that this channel is communicating to a low-speed device. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
15	epdir	Endpoint Direction (EPDir) Indicates whether the transaction is IN or OUT. 1'b0: OUT 1'b1: IN **Value** **Description** 0x0 OUT 0x1 IN	RW	0x0

Bit	Name	Description	Access	Reset
14:11	epnum	Endpoint Number (EPNum) Indicates the endpoint number on the device serving as the data source or sink. Value — Description 0xa — ENDPT10 0xb — ENDPT11 0xc — ENDPT12 0xd — ENDPT13 0xe — ENDPT14 0xf — ENDPT15 0x0 — ENDPT0 0x1 — ENDPT1 0x2 — ENDPT2 0x3 — ENDPT3 0x4 — ENDPT4 0x5 — ENDPT5 0x6 — ENDPT6 0x7 — ENDPT7 0x8 — ENDPT8 0x9 — ENDPT9	RW	0x0
10:0	mps	Maximum Packet Size (MPS) Indicates the maximum packet size of the associated endpoint.	RW	0x0

hcsplt7

Host Channel 7 Split Control Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB005E4
i_usbotg_1_hostgrp	0xFFB40400	0xFFB405E4

Offset: 0x1E4

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
spltena RW 0x0	Reserved														compsplt RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
xactpos RW 0x0		hubaddr RW 0x0							prtaddr RW 0x0						

hcsplt7 Fields

Bit	Name	Description	Access	Reset
31	spltena	Split Enable (SpltEna) The application sets this field to indicate that this channel is enabled to perform split transactions. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
16	compsplt	Do Complete Split (CompSplt) The application sets this field to request the OTG host to perform a complete split transaction. **Value** — **Description** 0x0 — NOSPLIT 0x1 — SPLIT	RW	0x0

Bit	Name	Description	Access	Reset
15:14	xactpos	Transaction Position (XactPos) This field is used to determine whether to send all, first, middle, or last payloads with each OUT transaction. 2'b11: All. This is the entire data payload is of this transaction (which is less than or equal to 188 bytes). 2'b10: Begin. This is the first data payload of this transaction (which is larger than 188 bytes). 2'b00: Mid. This is the middle payload of this transaction (which is larger than 188 bytes). 2'b01: End. This is the last payload of this transaction (which is larger than 188 bytes). **Value** **Description** 0x0 MIDDLE 0x1 END 0x2 BEGIN 0x3 ALL	RW	0x0
13:7	hubaddr	Hub Address (HubAddr) This field holds the device address of the transaction translator's hub.	RW	0x0
6:0	prtaddr	Port Address (PrtAddr) This field is the port number of the recipient transaction translator.	RW	0x0

hcint7

Host Channel 7 Interrupt Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB005E8

Module Instance	Base Address	Register Address
i_usbotg_1_hostgrp	0xFFB40400	0xFFB405E8

Offset: 0x1E8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		desc_lst_rollintr RW 0x0	xcs_xact_err RW 0x0	bnaintr RW 0x0	datatglerr RW 0x0	frmovrun RW 0x0	bblerr RW 0x0	xacterr RW 0x0	nyet RW 0x0	ack RW 0x0	nak RW 0x0	stall RW 0x0	ahberr RW 0x0	chhltd RW 0x0	xfercompl RW 0x0

hcint7 Fields

Bit	Name	Description	Access	Reset
13	desc_lst_rollintr	Descriptor rollover interrupt (DESC_LST_ROLLIntr) This bit is valid only when Scatter/Gather DMA mode is enabled. The core sets this bit when the corresponding channel's descriptor list rolls over. For non Scatter/Gather DMA mode, this bit is reserved. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
12	xcs_xact_err	Excessive Transaction Error (XCS_XACT_ERR) This bit is valid only when Scatter/Gather DMA mode is enabled. The core sets this bit when 3 consecutive transaction errors occurred on the USB bus. XCS_XACT_ERR will not be generated for Isochronous channels. For non Scatter/Gather DMA mode, this bit is reserved. **Value** **Description** 0x0 INACTIVE 0x1 ACVTIVE	RW	0x0
11	bnaintr	BNA (Buffer Not Available) Interrupt (BNAIntr) This bit is valid only when Scatter/Gather DMA mode is enabled. The core generates this interrupt when the descriptor accessed is not ready for the Core to process. BNA will not be generated for Isochronous channels. For non Scatter/Gather DMA mode, this bit is reserved. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
10	datatglerr	Data Toggle Error (DataTglErr).This bit can be set only by the core and the application should write 1 to clear it.In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
9	frmovrun	Frame Overrun (FrmOvrun).In Scatter/ Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the applica- tion should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
8	bblerr	Babble Error (BblErr) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core..This bit can be set only by the core and the applica- tion should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
7	xacterr	Transaction Error (XactErr) Indicates one of the following errors occurred on the USB. CRC check failure Timeout Bit stuff error False EOP In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the applica- tion should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
6	nyet	NYET Response Received Interrupt (NYET) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
5	ack	ACK Response Received/Transmitted Interrupt (ACK) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
4	nak	NAK Response Received Interrupt (NAK) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
3	stall	STALL Response Received Interrupt (STALL) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
2	ahberr	AHB Error (AHBErr) This is generated only in Internal DMA mode when there is an AHB error during AHB read/write. The application can read the corresponding channel's DMA address register to get the error address. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
1	chhltd	Channel Halted (ChHltd) In non Scatter/Gather DMA mode, it indicates the transfer completed abnormally either because of any USB transaction error or in response to disable request by the application or because of a completed transfer. in Scatter/gather DMA mode, this indicates that transfer completed due to any of the following . EOL being set in descriptor . AHB error . Excessive transaction errors . Babble . Stall **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
0	xfercompl	Transfer Completed (XferCompl) Transfer completed normally without any errors.This bit can be set only by the core and the application should write 1 to clear it. For Scatter/Gather DMA mode, it indicates that current descriptor processing got completed with IOC bit set in its descriptor. In non Scatter/Gather DMA mode, it indicates that Transfer completed normally without any errors. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

hcintmsk7

Host Channel 7 Interrupt Mask Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB005EC
i_usbotg_1_hostgrp	0xFFB40400	0xFFB405EC

Offset: 0x1EC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		frm_lst_rollintrmsk RW 0x0	Reserved	bnaintrmsk RW 0x0	datatglerrmsk RW 0x0	frmovrunmsk RW 0x0	bblerrmsk RW 0x0	xacterrmsk RW 0x0	nyetmsk RW 0x0	ackmsk RW 0x0	nakmsk RW 0x0	stallmsk RW 0x0	ahberrmsk RW 0x0	chhltdmsk RW 0x0	xfercomplmsk RW 0x0

hcintmsk7 Fields

Bit	Name	Description	Access	Reset
13	frm_lst_rollintrmsk	Framelist rollover interrupt Mask register(FRM_LST_ROLLIntrMsk) This bit is valid only when Scatter/Gather DMA mode is enabled. **Value** — **Description** 0x0 — MASK 0x1 — NOMASK	RW	0x0

Bit	Name	Description	Access	Reset
11	bnaintrmsk	BNA (Buffer Not Available) Interrupt mask register (BNAIntrMsk) This bit is valid only when Scatter/Gather DMA mode is enabled. **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0
10	datatglerrmsk	Data Toggle Error Mask (DataTglErrMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
9	frmovrunmsk	Frame Overrun Mask (FrmOvrunMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
8	bblerrmsk	Babble Error Mask (BblErrMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
7	xacterrmsk	Transaction Error Mask (XactErrMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0

Bit	Name	Description	Access	Reset
6	nyetmsk	NYET Response Received Interrupt Mask (NyetMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
5	ackmsk	ACK Response Received/Transmitted Interrupt Mask (AckMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
4	nakmsk	NAK Response Received Interrupt Mask (NakMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
3	stallmsk	STALL Response Received Interrupt Mask (StallMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0

Bit	Name	Description	Access	Reset
2	ahberrmsk	AHB Error Mask (AHBErrMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn. **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0
1	chhltdmsk	Channel Halted Mask (ChHltdMsk) **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0
0	xfercomplmsk	Transfer Completed Mask (XferComplMsk) **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0

hctsiz7

Host Channel 7 Transfer Size Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB005F0
i_usbotg_1_hostgrp	0xFFB40400	0xFFB405F0

Offset: 0x1F0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dopng RW 0x0	pid RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
xfersize RW 0x0															

hctsiz7 Fields

Bit	Name	Description	Access	Reset
31	dopng	Do Ping (DoPng) This bit is used only For OUT transfers. Setting this field to 1 directs the host to do PING protocol. Note: Do not Set this bit For IN transfers. If this bit is Set For for IN transfers it disables the channel. **Value** — **Description** 0x0 — NOPING 0x1 — PING	RW	0x0

Bit	Name	Description	Access	Reset
30:29	pid	PID (Pid) The application programs this field with the type of PID to use For the initial transaction. The host maintains this field For the rest of the transfer. 2'b00: DATA0 2'b01: DATA2 2'b10: DATA1 2'b11: MDATA (non-control)/SETUP (control) **Value** **Description** 0x0 DATA0 0x1 DATA2 0x2 DATA1 0x3 MDATA	RW	0x0
28:19	pktcnt	Packet Count (PktCnt) This field is programmed by the application with the expected number of packets to be transmitted (OUT) or received (IN). The host decrements this count on every successful transmission or reception of an OUT/IN packet. Once this count reaches zero, the application is interrupted to indicate normal completion. The width of this counter is specified as Width of Packet Counters	RW	0x0
18:0	xfersize	Transfer Size (XferSize) For an OUT, this field is the number of data bytes the host sends during the transfer. For an IN, this field is the buffer size that the application has Reserved For the transfer. The application is expected to program this field as an integer multiple of the maximum packet size For IN transactions (periodic and non-periodic). The width of this counter is specified as Width of Transfer Size Counters	RW	0x0

hcdma7

Host Channel 7 DMA Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB005F4
i_usbotg_1_hostgrp	0xFFB40400	0xFFB405F4

Offset: 0x1F4

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcdma7 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
hcdma7 RW 0x0															

hcdma7 Fields

Bit	Name	Description	Access	Reset
31:0	hcdma7	Buffer DMA Mode: [31:0] DMA Address (DMAAddr) This field holds the start address in the external memory from which the data for the endpoint must be fetched or to which it must be stored. This register is incremented on every AHB transac- tion. Scatter-Gather DMA (DescDMA) Mode: [31:9] (Non Isoc) Non-Isochronous: [31:N] (Isoc) Isochronous: This field holds the start address of the 512 bytes page. The first descriptor in the list should be located in this address. The first	RW	0x0

Bit	Name	Description	Access	Reset
		descriptor may be or may not be ready. The core starts processing the list from the CTD value. This field holds the address of the 2*(nTD+1) bytes of locations in which the isochronous descriptors are present where N is based on nTD as per Table below [31:N] Base Address [N-1:3] Offset [2:0] 000 HS ISOC nTD N 7 6 15 7 31 8 63 9 127 10 255 11 FS ISOC nTD N 1 4 3 5 7 6 15 7 31 8 63 9 [N-1:3] (Isoc): [8:3] (Non Isoc): Current Transfer Desc(CTD): Non Isochronous: This value is in terms of number of descriptors. The values can be from 0 to 63. 0 - 1 descriptor. 63 - 64 descriptors. This field indicates the current descriptor processed in the list. This field is updated both by application and the core. For example, if the application enables the channel after programming CTD=5, then the core will start processing the 6th descriptor. The address is obtained		

Bit	Name	Description	Access	Reset
		by adding a value of (8bytes*5=) 40(decimal) to DMAAddr. Isochronous: CTD for isochronous is based on the current frame/(micro)frame value. Need to be set to zero by application.Scatter-Gather DMA (DescDMA) Mode: [31:9] (Non Isoc) Non-Isochronous: [31:N] (Isoc) Isochronous: This field holds the start address of the 512 bytes page. The first descriptor in the list should be located in this address. The first descriptor may be or may not be ready. The core starts processing the list from the CTD value. This field holds the address of the 2*(nTD+1) bytes of locations in which the isochronous descriptors are present where N is based on nTD as per Table below [31:N] Base Address [N-1:3] Offset [2:0] 000 HS ISOC nTD N 7 6 15 7 31 8 63 9 127 10 255 11 FS ISOC nTD N 1 4 3 5 7 6 15 7 31 8 63 9 [N-1:3] (Isoc): [8:3] (Non Isoc): Current Transfer Desc(CTD):		

Bit	Name	Description	Access	Reset
		Non Isochronous: This value is in terms of number of descriptors. The values can be from 0 to 63. 0 - 1 descriptor. 63 - 64 descriptors. This field indicates the current descriptor processed in the list. This field is updated both by application and the core. For example, if the application enables the channel after programming CTD=5, then the core will start processing the 6th descriptor. The address is obtained by adding a value of (8bytes*5=) 40(decimal) to DMAAddr. Isochronous: CTD for isochronous is based on the current frame/(micro)frame value. Need to be set to zero by application.		

hcdmab7

Host Channel 7 DMA Buffer Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB005FC
i_usbotg_1_hostgrp	0xFFB40400	0xFFB405FC

Offset: 0x1FC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcdmab7 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
hcdmab7 RW 0x0															

hcdmab7 Fields

Bit	Name	Description	Access	Reset
31:0	hcdmab7	Holds the current buffer address. This register is updated as and when the data transfer for the corresponding end point is in progress. This register is present only in Scatter/Gather DMA mode. Otherwise this field is reserved.	RW	0x0

hcchar8

Host Channel 8 Characteristics Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB00600
i_usbotg_1_hostgrp	0xFFB40400	0xFFB40600

Offset: 0x200

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
chena RW 0x0	chdis RW 0x0	oddfrm RW 0x0	devaddr RW 0x0							ec RW 0x0		eptype RW 0x0		lspddev RW 0x0	Reserved
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
epdir RW 0x0	epnum RW 0x0				mps RW 0x0										

hcchar8 Fields

Bit	Name	Description	Access	Reset
31	chena	Channel Enable (ChEna) When Scatter/Gather mode is enabled 1'b0: Indicates that the descriptor structure is not yet ready. 1'b1: Indicates that the descriptor structure and data buffer with data is setup and this channel can access the descriptor. When Scatter/Gather mode is disabled This field is set by the application and cleared by the OTG host. 1'b0: Channel disabled 1'b1: Channel enabled **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
30	chdis	Channel Disable (ChDis) The application sets this bit to stop transmitting/receiving data on a channel, even before the transfer For that channel is complete. The application must wait For the Channel Disabled interrupt before treating the channel as disabled. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
29	oddfrm	Odd Frame (OddFrm) This field is set (reset) by the application to indicate that the OTG host must perform a transfer in an odd (micro)frame. This field is applicable for only periodic (isochronous and interrupt) transactions. 1'b0: Even (micro)frame 1'b1: Odd (micro)frame	RW	0x0
28:22	devaddr	Device Address (DevAddr) This field selects the specific device serving as the data source or sink.	RW	0x0

Bit	Name	Description	Access	Reset
21:20	ec	Multi Count (MC) / Error Count (EC) When the Split Enable bit of the Host Channel-n Split Control register (HCSPLTn.SpltEna) is reset (1'b0), this field indicates to the host the number of transactions that must be executed per microframe For this periodic endpoint. For non periodic transfers, this field is used only in DMA mode, and specifies the number packets to be fetched For this channel before the internal DMA engine changes arbitration. 2'b00: Reserved This field yields undefined results. 2'b01: 1 transaction 2'b10: 2 transactions to be issued For this endpoint per microframe 2'b11: 3 transactions to be issued For this endpoint per microframe When HCSPLTn.SpltEna is Set (1'b1), this field indicates the number of immediate retries to be performed For a periodic split transactions on transaction errors. This field must be Set to at least 2'b01. **Value** **Description** 0x0 RESERVED 0x1 TRANSONE 0x2 TRANSTWO 0x3 TRANSTHREE	RW	0x0

Bit	Name	Description	Access	Reset
19:18	eptype	Endpoint Type (EPType) Indicates the transfer type selected. 2'b00: Control 2'b01: Isochronous 2'b10: Bulk 2'b11: Interrupt **Value** **Description** 0x0 CTRL 0x1 ISOC 0x2 BULK 0x3 INTERR	RW	0x0
17	lspddev	Low-Speed Device (LSpdDev) This field is Set by the application to indicate that this channel is communicating to a low-speed device. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
15	epdir	Endpoint Direction (EPDir) Indicates whether the transaction is IN or OUT. 1'b0: OUT 1'b1: IN **Value** **Description** 0x0 OUT 0x1 IN	RW	0x0

Bit	Name	Description	Access	Reset
14:11	epnum	Endpoint Number (EPNum) Indicates the endpoint number on the device serving as the data source or sink. Value — Description 0xa — ENDPT10 0xb — ENDPT11 0xc — ENDPT12 0xd — ENDPT13 0xe — ENDPT14 0xf — ENDPT15 0x0 — ENDPT0 0x1 — ENDPT1 0x2 — ENDPT2 0x3 — ENDPT3 0x4 — ENDPT4 0x5 — ENDPT5 0x6 — ENDPT6 0x7 — ENDPT7 0x8 — ENDPT8 0x9 — ENDPT9	RW	0x0
10:0	mps	Maximum Packet Size (MPS) Indicates the maximum packet size of the associated endpoint.	RW	0x0

hcsplt8

Host Channel 8 Split Control Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB00604
i_usbotg_1_hostgrp	0xFFB40400	0xFFB40604

Offset: 0x204

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
spltena RW 0x0	Reserved														compsplt RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
xactpos RW 0x0	hubaddr RW 0x0								prtaddr RW 0x0						

hcsplt8 Fields

Bit	Name	Description	Access	Reset
31	spltena	Split Enable (SpltEna) The application sets this field to indicate that this channel is enabled to perform split transactions. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
16	compsplt	Do Complete Split (CompSplt) The application sets this field to request the OTG host to perform a complete split transaction. **Value** — **Description** 0x0 — NOSPLIT 0x1 — SPLIT	RW	0x0

Bit	Name	Description	Access	Reset
15:14	xactpos	Transaction Position (XactPos) This field is used to determine whether to send all, first, middle, or last payloads with each OUT transaction. 2'b11: All. This is the entire data payload is of this transaction (which is less than or equal to 188 bytes). 2'b10: Begin. This is the first data payload of this transaction (which is larger than 188 bytes). 2'b00: Mid. This is the middle payload of this transaction (which is larger than 188 bytes). 2'b01: End. This is the last payload of this transaction (which is larger than 188 bytes). **Value** **Description** 0x0 MIDDLE 0x1 END 0x2 BEGIN 0x3 ALL	RW	0x0
13:7	hubaddr	Hub Address (HubAddr) This field holds the device address of the transaction translator's hub.	RW	0x0
6:0	prtaddr	Port Address (PrtAddr) This field is the port number of the recipient transaction translator.	RW	0x0

hcint8

Host Channel 8 Interrupt Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB00608

Module Instance	Base Address	Register Address
i_usbotg_1_hostgrp	0xFFB40400	0xFFB40608

Offset: 0x208

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		desc_lst_rollintr RW 0x0	xcs_xact_err RW 0x0	bnaintr RW 0x0	datatglerr RW 0x0	frmovrun RW 0x0	bblerr RW 0x0	xacterr RW 0x0	nyet RW 0x0	ack RW 0x0	nak RW 0x0	stall RW 0x0	ahberr RW 0x0	chhltd RW 0x0	xfercompl RW 0x0

hcint8 Fields

Bit	Name	Description	Access	Reset
13	desc_lst_rollintr	Descriptor rollover interrupt (DESC_LST_ROLLIntr) This bit is valid only when Scatter/Gather DMA mode is enabled. The core sets this bit when the corresponding channel's descriptor list rolls over. For non Scatter/Gather DMA mode, this bit is reserved. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
12	xcs_xact_err	Excessive Transaction Error (XCS_XACT_ERR) This bit is valid only when Scatter/Gather DMA mode is enabled. The core sets this bit when 3 consecutive transaction errors occurred on the USB bus. XCS_XACT_ERR will not be generated for Isochronous channels. For non Scatter/Gather DMA mode, this bit is reserved. **Value** **Description** 0x0 INACTIVE 0x1 ACVTIVE	RW	0x0
11	bnaintr	BNA (Buffer Not Available) Interrupt (BNAIntr) This bit is valid only when Scatter/Gather DMA mode is enabled. The core generates this interrupt when the descriptor accessed is not ready for the Core to process. BNA will not be generated for Isochronous channels. For non Scatter/Gather DMA mode, this bit is reserved. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
10	datatglerr	Data Toggle Error (DataTglErr).This bit can be set only by the core and the application should write 1 to clear it.In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
9	frmovrun	Frame Overrun (FrmOvrun).In Scatter/ Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the applica- tion should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
8	bblerr	Babble Error (BblErr) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core..This bit can be set only by the core and the applica- tion should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
7	xacterr	Transaction Error (XactErr) Indicates one of the following errors occurred on the USB. CRC check failure Timeout Bit stuff error False EOP In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the applica- tion should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
6	nyet	NYET Response Received Interrupt (NYET) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
5	ack	ACK Response Received/Transmitted Interrupt (ACK) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
4	nak	NAK Response Received Interrupt (NAK) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
3	stall	STALL Response Received Interrupt (STALL) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
2	ahberr	AHB Error (AHBErr) This is generated only in Internal DMA mode when there is an AHB error during AHB read/write. The application can read the corresponding channel's DMA address register to get the error address. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
1	chhltd	Channel Halted (ChHltd) In non Scatter/Gather DMA mode, it indicates the transfer completed abnormally either because of any USB transaction error or in response to disable request by the application or because of a completed transfer. in Scatter/gather DMA mode, this indicates that transfer completed due to any of the following . EOL being set in descriptor . AHB error . Excessive transaction errors . Babble . Stall **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
0	xfercompl	Transfer Completed (XferCompl) Transfer completed normally without any errors.This bit can be set only by the core and the application should write 1 to clear it. For Scatter/Gather DMA mode, it indicates that current descriptor processing got completed with IOC bit set in its descriptor. In non Scatter/Gather DMA mode, it indicates that Transfer completed normally without any errors. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

hcintmsk8

Host Channel 8 Interrupt Mask Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB0060C
i_usbotg_1_hostgrp	0xFFB40400	0xFFB4060C

Offset: 0x20C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		frm_lst_rollintrmsk RW 0x0	Reserved	bnaintrmsk RW 0x0	datatglerrmsk RW 0x0	frmovrunmsk RW 0x0	bblerrmsk RW 0x0	xacterrmsk RW 0x0	nyetmsk RW 0x0	ackmsk RW 0x0	nakmsk RW 0x0	stallmsk RW 0x0	ahberrmsk RW 0x0	chhltdmsk RW 0x0	xfercomplmsk RW 0x0

hcintmsk8 Fields

Bit	Name	Description	Access	Reset
13	frm_lst_rollintrmsk	Framelist rollover interrupt Mask register(FRM_LST_ROLLIntrMsk) This bit is valid only when Scatter/Gather DMA mode is enabled. **Value** — **Description** 0x0 — MASK 0x1 — NOMASK	RW	0x0

Bit	Name	Description	Access	Reset
11	bnaintrmsk	BNA (Buffer Not Available) Interrupt mask register (BNAIntrMsk) This bit is valid only when Scatter/Gather DMA mode is enabled. **Value** — **Description** 0x0 — MASK 0x1 — NOMASK	RW	0x0
10	datatglerrmsk	Data Toggle Error Mask (DataTglErrMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
9	frmovrunmsk	Frame Overrun Mask (FrmOvrunMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
8	bblerrmsk	Babble Error Mask (BblErrMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
7	xacterrmsk	Transaction Error Mask (XactErrMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0

Bit	Name	Description	Access	Reset
6	nyetmsk	NYET Response Received Interrupt Mask (NyetMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
5	ackmsk	ACK Response Received/Transmitted Interrupt Mask (AckMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
4	nakmsk	NAK Response Received Interrupt Mask (NakMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
3	stallmsk	STALL Response Received Interrupt Mask (StallMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0

Bit	Name	Description	Access	Reset
2	ahberrmsk	AHB Error Mask (AHBErrMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn. **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0
1	chhltdmsk	Channel Halted Mask (ChHltdMsk) **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0
0	xfercomplmsk	Transfer Completed Mask (XferComplMsk) **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0

hctsiz8

Host Channel 8 Transfer Size Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB00610
i_usbotg_1_hostgrp	0xFFB40400	0xFFB40610

Offset: 0x210

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dopng RW 0x0	pid RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
xfersize RW 0x0															

hctsiz8 Fields

Bit	Name	Description	Access	Reset
31	dopng	Do Ping (DoPng) This bit is used only For OUT transfers. Setting this field to 1 directs the host to do PING protocol. Note: Do not Set this bit For IN transfers. If this bit is Set For for IN transfers it disables the channel. **Value** — **Description** 0x0 — NOPING 0x1 — PING	RW	0x0

Bit	Name	Description	Access	Reset
30:29	pid	PID (Pid) The application programs this field with the type of PID to use For the initial transaction. The host maintains this field For the rest of the transfer. 2'b00: DATA0 2'b01: DATA2 2'b10: DATA1 2'b11: MDATA (non-control)/SETUP (control) **Value** **Description** 0x0 DATA0 0x1 DATA2 0x2 DATA1 0x3 MDATA	RW	0x0
28:19	pktcnt	Packet Count (PktCnt) This field is programmed by the application with the expected number of packets to be transmitted (OUT) or received (IN). The host decrements this count on every successful transmission or reception of an OUT/IN packet. Once this count reaches zero, the application is interrupted to indicate normal completion. The width of this counter is specified as Width of Packet Counters	RW	0x0
18:0	xfersize	Transfer Size (XferSize) For an OUT, this field is the number of data bytes the host sends during the transfer. For an IN, this field is the buffer size that the application has Reserved For the transfer. The application is expected to program this field as an integer multiple of the maximum packet size For IN transactions (periodic and non-periodic). The width of this counter is specified as Width of Transfer Size Counters	RW	0x0

hcdma8

Host Channel 8 DMA Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB00614
i_usbotg_1_hostgrp	0xFFB40400	0xFFB40614

Offset: 0x214

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcdma8 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
hcdma8 RW 0x0															

hcdma8 Fields

Bit	Name	Description	Access	Reset
31:0	hcdma8	Buffer DMA Mode: [31:0] DMA Address (DMAAddr) This field holds the start address in the external memory from which the data for the endpoint must be fetched or to which it must be stored. This register is incremented on every AHB transac- tion. Scatter-Gather DMA (DescDMA) Mode: [31:9] (Non Isoc) Non-Isochronous: [31:N] (Isoc) Isochronous: This field holds the start address of the 512 bytes page. The first descriptor in the list should be located in this address. The first	RW	0x0

Bit	Name	Description	Access	Reset
		descriptor may be or may not be ready. The core starts processing the list from the CTD value. This field holds the address of the 2*(nTD+1) bytes of locations in which the isochronous descriptors are present where N is based on nTD as per Table below [31:N] Base Address [N-1:3] Offset [2:0] 000 HS ISOC nTD N 7 6 15 7 31 8 63 9 127 10 255 11 FS ISOC nTD N 1 4 3 5 7 6 15 7 31 8 63 9 [N-1:3] (Isoc): [8:3] (Non Isoc): Current Transfer Desc(CTD): Non Isochronous: This value is in terms of number of descriptors. The values can be from 0 to 63. 0 - 1 descriptor. 63 - 64 descriptors. This field indicates the current descriptor processed in the list. This field is updated both by application and the core. For example, if the application enables the channel after programming CTD=5, then the core will start processing the 6th descriptor. The address is obtained		

Bit	Name	Description	Access	Reset
		by adding a value of (8bytes*5=) 40(decimal) to DMAAddr. Isochronous: CTD for isochronous is based on the current frame/(micro)frame value. Need to be set to zero by application.Scatter-Gather DMA (DescDMA) Mode: [31:9] (Non Isoc) Non-Isochronous: [31:N] (Isoc) Isochronous: This field holds the start address of the 512 bytes page. The first descriptor in the list should be located in this address. The first descriptor may be or may not be ready. The core starts processing the list from the CTD value. This field holds the address of the 2*(nTD+1) bytes of locations in which the isochronous descriptors are present where N is based on nTD as per Table below [31:N] Base Address [N-1:3] Offset [2:0] 000 HS ISOC nTD N 7 6 15 7 31 8 63 9 127 10 255 11 FS ISOC nTD N 1 4 3 5 7 6 15 7 31 8 63 9 [N-1:3] (Isoc): [8:3] (Non Isoc): Current Transfer Desc(CTD):		

Bit	Name	Description	Access	Reset
		Non Isochronous: This value is in terms of number of descriptors. The values can be from 0 to 63. 0 - 1 descriptor. 63 - 64 descriptors. This field indicates the current descriptor processed in the list. This field is updated both by application and the core. For example, if the application enables the channel after programming CTD=5, then the core will start processing the 6th descriptor. The address is obtained by adding a value of (8bytes*5=) 40(decimal) to DMAAddr. Isochronous: CTD for isochronous is based on the current frame/(micro)frame value. Need to be set to zero by application.		

hcdmab8

Host Channel 8 DMA Buffer Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB0061C
i_usbotg_1_hostgrp	0xFFB40400	0xFFB4061C

Offset: 0x21C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcdmab8 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
hcdmab8 RW 0x0															

hcdmab8 Fields

Bit	Name	Description	Access	Reset
31:0	hcdmab8	Holds the current buffer address. This register is updated as and when the data transfer for the corresponding end point is in progress. This register is present only in Scatter/Gather DMA mode. Otherwise this field is reserved.	RW	0x0

hcchar9

Host Channel 9 Characteristics Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB00620
i_usbotg_1_hostgrp	0xFFB40400	0xFFB40620

Offset: 0x220

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
chena RW 0x0	chdis RW 0x0	oddfrm RW 0x0	devaddr RW 0x0							ec RW 0x0		eptype RW 0x0		lspddev RW 0x0	Reserved
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
epdir RW 0x0	epnum RW 0x0				mps RW 0x0										

hcchar9 Fields

Bit	Name	Description	Access	Reset
31	chena	Channel Enable (ChEna) When Scatter/Gather mode is enabled 1'b0: Indicates that the descriptor structure is not yet ready. 1'b1: Indicates that the descriptor structure and data buffer with data is setup and this channel can access the descriptor. When Scatter/Gather mode is disabled This field is set by the application and cleared by the OTG host. 1'b0: Channel disabled 1'b1: Channel enabled **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
30	chdis	Channel Disable (ChDis) The application sets this bit to stop transmitting/receiving data on a channel, even before the transfer For that channel is complete. The application must wait For the Channel Disabled interrupt before treating the channel as disabled. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
29	oddfrm	Odd Frame (OddFrm) This field is set (reset) by the application to indicate that the OTG host must perform a transfer in an odd (micro)frame. This field is applicable for only periodic (isochronous and interrupt) transactions. 1'b0: Even (micro)frame 1'b1: Odd (micro)frame	RW	0x0
28:22	devaddr	Device Address (DevAddr) This field selects the specific device serving as the data source or sink.	RW	0x0

Bit	Name	Description	Access	Reset
21:20	ec	Multi Count (MC) / Error Count (EC) When the Split Enable bit of the Host Channel-n Split Control register (HCSPLTn.SpltEna) is reset (1'b0), this field indicates to the host the number of transactions that must be executed per microframe For this periodic endpoint. For non periodic transfers, this field is used only in DMA mode, and specifies the number packets to be fetched For this channel before the internal DMA engine changes arbitration. 2'b00: Reserved This field yields undefined results. 2'b01: 1 transaction 2'b10: 2 transactions to be issued For this endpoint per microframe 2'b11: 3 transactions to be issued For this endpoint per microframe When HCSPLTn.SpltEna is Set (1'b1), this field indicates the number of immediate retries to be performed For a periodic split transactions on transaction errors. This field must be Set to at least 2'b01. **Value** **Description** 0x0 RESERVED 0x1 TRANSONE 0x2 TRANSTWO 0x3 TRANSTHREE	RW	0x0

Bit	Name	Description	Access	Reset
19:18	eptype	Endpoint Type (EPType) Indicates the transfer type selected. 2'b00: Control 2'b01: Isochronous 2'b10: Bulk 2'b11: Interrupt **Value** **Description** 0x0 CTRL 0x1 ISOC 0x2 BULK 0x3 INTERR	RW	0x0
17	lspddev	Low-Speed Device (LSpdDev) This field is Set by the application to indicate that this channel is communicating to a low-speed device. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
15	epdir	Endpoint Direction (EPDir) Indicates whether the transaction is IN or OUT. 1'b0: OUT 1'b1: IN **Value** **Description** 0x0 OUT 0x1 IN	RW	0x0

Bit	Name	Description	Access	Reset
14:11	epnum	Endpoint Number (EPNum) Indicates the endpoint number on the device serving as the data source or sink. Value — Description 0xa — ENDPT10 0xb — ENDPT11 0xc — ENDPT12 0xd — ENDPT13 0xe — ENDPT14 0xf — ENDPT15 0x0 — ENDPT0 0x1 — ENDPT1 0x2 — ENDPT2 0x3 — ENDPT3 0x4 — ENDPT4 0x5 — ENDPT5 0x6 — ENDPT6 0x7 — ENDPT7 0x8 — ENDPT8 0x9 — ENDPT9	RW	0x0
10:0	mps	Maximum Packet Size (MPS) Indicates the maximum packet size of the associated endpoint.	RW	0x0

hcsplt9

Host Channel 9 Split Control Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB00624
i_usbotg_1_hostgrp	0xFFB40400	0xFFB40624

Offset: 0x224

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
spltena RW 0x0	Reserved														compsplt RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
xactpos RW 0x0		hubaddr RW 0x0							prtaddr RW 0x0						

hcsplt9 Fields

Bit	Name	Description	Access	Reset
31	spltena	Split Enable (SpltEna) The application sets this field to indicate that this channel is enabled to perform split transactions. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
16	compsplt	Do Complete Split (CompSplt) The application sets this field to request the OTG host to perform a complete split transaction. **Value** — **Description** 0x0 — NOSPLIT 0x1 — SPLIT	RW	0x0

Bit	Name	Description	Access	Reset
15:14	xactpos	Transaction Position (XactPos) This field is used to determine whether to send all, first, middle, or last payloads with each OUT transaction. 2'b11: All. This is the entire data payload is of this transaction (which is less than or equal to 188 bytes). 2'b10: Begin. This is the first data payload of this transaction (which is larger than 188 bytes). 2'b00: Mid. This is the middle payload of this transaction (which is larger than 188 bytes). 2'b01: End. This is the last payload of this transaction (which is larger than 188 bytes). **Value** **Description** 0x0 MIDDLE 0x1 END 0x2 BEGIN 0x3 ALL	RW	0x0
13:7	hubaddr	Hub Address (HubAddr) This field holds the device address of the transaction translator's hub.	RW	0x0
6:0	prtaddr	Port Address (PrtAddr) This field is the port number of the recipient transaction translator.	RW	0x0

hcint9

Host Channel 9 Interrupt Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB00628

Module Instance	Base Address	Register Address
i_usbotg_1_hostgrp	0xFFB40400	0xFFB40628

Offset: 0x228

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		desc_lst_rollintr RW 0x0	xcs_xact_err RW 0x0	bnaintr RW 0x0	datatglerr RW 0x0	frmovrun RW 0x0	bblerr RW 0x0	xacterr RW 0x0	nyet RW 0x0	ack RW 0x0	nak RW 0x0	stall RW 0x0	ahberr RW 0x0	chhltd RW 0x0	xfercompl RW 0x0

hcint9 Fields

Bit	Name	Description	Access	Reset
13	desc_lst_rollintr	Descriptor rollover interrupt (DESC_LST_ROLLIntr) This bit is valid only when Scatter/Gather DMA mode is enabled. The core sets this bit when the corresponding channel's descriptor list rolls over. For non Scatter/Gather DMA mode, this bit is reserved. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
12	xcs_xact_err	Excessive Transaction Error (XCS_XACT_ERR) This bit is valid only when Scatter/Gather DMA mode is enabled. The core sets this bit when 3 consecutive transaction errors occurred on the USB bus. XCS_XACT_ERR will not be generated for Isochronous channels. For non Scatter/Gather DMA mode, this bit is reserved. **Value** **Description** 0x0 INACTIVE 0x1 ACVTIVE	RW	0x0
11	bnaintr	BNA (Buffer Not Available) Interrupt (BNAIntr) This bit is valid only when Scatter/Gather DMA mode is enabled. The core generates this interrupt when the descriptor accessed is not ready for the Core to process. BNA will not be generated for Isochronous channels. For non Scatter/Gather DMA mode, this bit is reserved. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
10	datatglerr	Data Toggle Error (DataTglErr).This bit can be set only by the core and the application should write 1 to clear it.In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
9	frmovrun	Frame Overrun (FrmOvrun).In Scatter/ Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the applica- tion should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
8	bblerr	Babble Error (BblErr) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core..This bit can be set only by the core and the applica- tion should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
7	xacterr	Transaction Error (XactErr) Indicates one of the following errors occurred on the USB. CRC check failure Timeout Bit stuff error False EOP In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the applica- tion should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
6	nyet	NYET Response Received Interrupt (NYET) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
5	ack	ACK Response Received/Transmitted Interrupt (ACK) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
4	nak	NAK Response Received Interrupt (NAK) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
3	stall	STALL Response Received Interrupt (STALL) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
2	ahberr	AHB Error (AHBErr) This is generated only in Internal DMA mode when there is an AHB error during AHB read/write. The application can read the corresponding channel's DMA address register to get the error address. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
1	chhltd	Channel Halted (ChHltd) In non Scatter/Gather DMA mode, it indicates the transfer completed abnormally either because of any USB transaction error or in response to disable request by the application or because of a completed transfer. in Scatter/gather DMA mode, this indicates that transfer completed due to any of the following . EOL being set in descriptor . AHB error . Excessive transaction errors . Babble . Stall **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
0	xfercompl	Transfer Completed (XferCompl) Transfer completed normally without any errors.This bit can be set only by the core and the application should write 1 to clear it. For Scatter/Gather DMA mode, it indicates that current descriptor processing got completed with IOC bit set in its descriptor. In non Scatter/Gather DMA mode, it indicates that Transfer completed normally without any errors. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

hcintmsk9

Host Channel 9 Interrupt Mask Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB0062C
i_usbotg_1_hostgrp	0xFFB40400	0xFFB4062C

Offset: 0x22C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		frm_lst_rollintrmsk RW 0x0	Reserved	bnaintrmsk RW 0x0	datatglerrmsk RW 0x0	frmovrunmsk RW 0x0	bblerrmsk RW 0x0	xacterrmsk RW 0x0	nyetmsk RW 0x0	ackmsk RW 0x0	nakmsk RW 0x0	stallmsk RW 0x0	ahberrmsk RW 0x0	chhltdmsk RW 0x0	xfercomplmsk RW 0x0

hcintmsk9 Fields

Bit	Name	Description	Access	Reset
13	frm_lst_rollintrmsk	Framelist rollover interrupt Mask register(FRM_LST_ROLLIntrMsk) This bit is valid only when Scatter/Gather DMA mode is enabled. Value: Description 0x0: MASK 0x1: NOMASK	RW	0x0

Bit	Name	Description	Access	Reset
11	bnaintrmsk	BNA (Buffer Not Available) Interrupt mask register (BNAIntrMsk) This bit is valid only when Scatter/Gather DMA mode is enabled. Value: 0x0 Description: MASK Value: 0x1 Description: NOMASK	RW	0x0
10	datatglerrmsk	Data Toggle Error Mask (DataTglErrMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
9	frmovrunmsk	Frame Overrun Mask (FrmOvrunMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
8	bblerrmsk	Babble Error Mask (BblErrMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
7	xacterrmsk	Transaction Error Mask (XactErrMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0

Bit	Name	Description	Access	Reset
6	nyetmsk	NYET Response Received Interrupt Mask (NyetMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
5	ackmsk	ACK Response Received/Transmitted Interrupt Mask (AckMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
4	nakmsk	NAK Response Received Interrupt Mask (NakMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
3	stallmsk	STALL Response Received Interrupt Mask (StallMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0

Bit	Name	Description	Access	Reset
2	ahberrmsk	AHB Error Mask (AHBErrMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn. **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0
1	chhltdmsk	Channel Halted Mask (ChHltdMsk) **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0
0	xfercomplmsk	Transfer Completed Mask (XferComplMsk) **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0

hctsiz9

Host Channel 9 Transfer Size Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB00630
i_usbotg_1_hostgrp	0xFFB40400	0xFFB40630

Offset: 0x230

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dopng RW 0x0	pid RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
xfersize RW 0x0															

hctsiz9 Fields

Bit	Name	Description	Access	Reset
31	dopng	Do Ping (DoPng) This bit is used only For OUT transfers. Setting this field to 1 directs the host to do PING protocol. Note: Do not Set this bit For IN transfers. If this bit is Set For for IN transfers it disables the channel. **Value** — **Description** 0x0 — NOPING 0x1 — PING	RW	0x0

Bit	Name	Description	Access	Reset
30:29	pid	PID (Pid) The application programs this field with the type of PID to use For the initial transaction. The host maintains this field For the rest of the transfer. 2'b00: DATA0 2'b01: DATA2 2'b10: DATA1 2'b11: MDATA (non-control)/SETUP (control) **Value** **Description** 0x0 DATA0 0x1 DATA2 0x2 DATA1 0x3 MDATA	RW	0x0
28:19	pktcnt	Packet Count (PktCnt) This field is programmed by the application with the expected number of packets to be transmitted (OUT) or received (IN). The host decrements this count on every successful transmission or reception of an OUT/IN packet. Once this count reaches zero, the application is interrupted to indicate normal completion. The width of this counter is specified as Width of Packet Counters	RW	0x0
18:0	xfersize	Transfer Size (XferSize) For an OUT, this field is the number of data bytes the host sends during the transfer. For an IN, this field is the buffer size that the application has Reserved For the transfer. The application is expected to program this field as an integer multiple of the maximum packet size For IN transactions (periodic and non-periodic). The width of this counter is specified as Width of Transfer Size Counters	RW	0x0

hcdma9

Host Channel 9 DMA Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB00634
i_usbotg_1_hostgrp	0xFFB40400	0xFFB40634

Offset: 0x234

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcdma9 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
hcdma9 RW 0x0															

hcdma9 Fields

Bit	Name	Description	Access	Reset
31:0	hcdma9	Buffer DMA Mode: [31:0] DMA Address (DMAAddr) This field holds the start address in the external memory from which the data for the endpoint must be fetched or to which it must be stored. This register is incremented on every AHB transac- tion. Scatter-Gather DMA (DescDMA) Mode: [31:9] (Non Isoc) Non-Isochronous: [31:N] (Isoc) Isochronous: This field holds the start address of the 512 bytes page. The first descriptor in the list should be located in this address. The first	RW	0x0

Bit	Name	Description	Access	Reset
		descriptor may be or may not be ready. The core starts processing the list from the CTD value. This field holds the address of the 2*(nTD+1) bytes of locations in which the isochronous descriptors are present where N is based on nTD as per Table below [31:N] Base Address [N-1:3] Offset [2:0] 000 HS ISOC nTD N 7 6 15 7 31 8 63 9 127 10 255 11 FS ISOC nTD N 1 4 3 5 7 6 15 7 31 8 63 9 [N-1:3] (Isoc): [8:3] (Non Isoc): Current Transfer Desc(CTD): Non Isochronous: This value is in terms of number of descriptors. The values can be from 0 to 63. 0 - 1 descriptor. 63 - 64 descriptors. This field indicates the current descriptor processed in the list. This field is updated both by application and the core. For example, if the application enables the channel after programming CTD=5, then the core will start processing the 6th descriptor. The address is obtained		

Bit	Name	Description	Access	Reset
		by adding a value of (8bytes*5=) 40(decimal) to DMAAddr. Isochronous: CTD for isochronous is based on the current frame/(micro)frame value. Need to be set to zero by application.Scatter-Gather DMA (DescDMA) Mode: [31:9] (Non Isoc) Non-Isochronous: [31:N] (Isoc) Isochronous: This field holds the start address of the 512 bytes page. The first descriptor in the list should be located in this address. The first descriptor may be or may not be ready. The core starts processing the list from the CTD value. This field holds the address of the 2*(nTD+1) bytes of locations in which the isochronous descriptors are present where N is based on nTD as per Table below [31:N] Base Address [N-1:3] Offset [2:0] 000 HS ISOC nTD N 7 6 15 7 31 8 63 9 127 10 255 11 FS ISOC nTD N 1 4 3 5 7 6 15 7 31 8 63 9 [N-1:3] (Isoc): [8:3] (Non Isoc): Current Transfer Desc(CTD):		

Bit	Name	Description	Access	Reset
		Non Isochronous: This value is in terms of number of descriptors. The values can be from 0 to 63. 0 - 1 descriptor. 63 - 64 descriptors. This field indicates the current descriptor processed in the list. This field is updated both by application and the core. For example, if the application enables the channel after programming CTD=5, then the core will start processing the 6th descriptor. The address is obtained by adding a value of (8bytes*5=) 40(decimal) to DMAAddr. Isochronous: CTD for isochronous is based on the current frame/(micro)frame value. Need to be set to zero by application.		

hcdmab9

Host Channel 9 DMA Buffer Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB0063C
i_usbotg_1_hostgrp	0xFFB40400	0xFFB4063C

Offset: 0x23C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcdmab9 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
hcdmab9 RW 0x0															

hcdmab9 Fields

Bit	Name	Description	Access	Reset
31:0	hcdmab9	Holds the current buffer address. This register is updated as and when the data transfer for the corresponding end point is in progress. This register is present only in Scatter/Gather DMA mode. Otherwise this field is reserved.	RW	0x0

hcchar10

Host Channel 10 Characteristics Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB00640
i_usbotg_1_hostgrp	0xFFB40400	0xFFB40640

Offset: 0x240

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
chena RW 0x0	chdis RW 0x0	oddfrm RW 0x0	devaddr RW 0x0							ec RW 0x0		eptype RW 0x0		lspddev RW 0x0	Reserved
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
epdir RW 0x0	epnum RW 0x0				mps RW 0x0										

hcchar10 Fields

Bit	Name	Description	Access	Reset
31	chena	Channel Enable (ChEna) When Scatter/Gather mode is enabled 1'b0: Indicates that the descriptor structure is not yet ready. 1'b1: Indicates that the descriptor structure and data buffer with data is setup and this channel can access the descriptor. When Scatter/Gather mode is disabled This field is set by the application and cleared by the OTG host. 1'b0: Channel disabled 1'b1: Channel enabled **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
30	chdis	Channel Disable (ChDis) The application sets this bit to stop transmitting/receiving data on a channel, even before the transfer For that channel is complete. The application must wait For the Channel Disabled interrupt before treating the channel as disabled. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
29	oddfrm	Odd Frame (OddFrm) This field is set (reset) by the application to indicate that the OTG host must perform a transfer in an odd (micro)frame. This field is applicable for only periodic (isochronous and interrupt) transactions. 1'b0: Even (micro)frame 1'b1: Odd (micro)frame	RW	0x0
28:22	devaddr	Device Address (DevAddr) This field selects the specific device serving as the data source or sink.	RW	0x0

Bit	Name	Description	Access	Reset
21:20	ec	Multi Count (MC) / Error Count (EC) When the Split Enable bit of the Host Channel-n Split Control register (HCSPLTn.SpltEna) is reset (1'b0), this field indicates to the host the number of transactions that must be executed per microframe For this periodic endpoint. For non periodic transfers, this field is used only in DMA mode, and specifies the number packets to be fetched For this channel before the internal DMA engine changes arbitration. 2'b00: Reserved This field yields undefined results. 2'b01: 1 transaction 2'b10: 2 transactions to be issued For this endpoint per microframe 2'b11: 3 transactions to be issued For this endpoint per microframe When HCSPLTn.SpltEna is Set (1'b1), this field indicates the number of immediate retries to be performed For a periodic split transactions on transaction errors. This field must be Set to at least 2'b01. **Value** **Description** 0x0 RESERVED 0x1 TRANSONE 0x2 TRANSTWO 0x3 TRANSTHREE	RW	0x0

Bit	Name	Description	Access	Reset
19:18	eptype	Endpoint Type (EPType) Indicates the transfer type selected. 2'b00: Control 2'b01: Isochronous 2'b10: Bulk 2'b11: Interrupt **Value** — **Description** 0x0 — CTRL 0x1 — ISOC 0x2 — BULK 0x3 — INTERR	RW	0x0
17	lspddev	Low-Speed Device (LSpdDev) This field is Set by the application to indicate that this channel is communicating to a low-speed device. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
15	epdir	Endpoint Direction (EPDir) Indicates whether the transaction is IN or OUT. 1'b0: OUT 1'b1: IN **Value** — **Description** 0x0 — OUT 0x1 — IN	RW	0x0

Bit	Name	Description	Access	Reset
14:11	epnum	Endpoint Number (EPNum) Indicates the endpoint number on the device serving as the data source or sink. Value — Description 0xa — ENDPT10 0xb — ENDPT11 0xc — ENDPT12 0xd — ENDPT13 0xe — ENDPT14 0xf — ENDPT15 0x0 — ENDPT0 0x1 — ENDPT1 0x2 — ENDPT2 0x3 — ENDPT3 0x4 — ENDPT4 0x5 — ENDPT5 0x6 — ENDPT6 0x7 — ENDPT7 0x8 — ENDPT8 0x9 — ENDPT9	RW	0x0
10:0	mps	Maximum Packet Size (MPS) Indicates the maximum packet size of the associated endpoint.	RW	0x0

hcsplt10

Host Channel 10 Split Control Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB00644
i_usbotg_1_hostgrp	0xFFB40400	0xFFB40644

Offset: 0x244

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
spltena RW 0x0	Reserved														compsplt RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
xactpos RW 0x0	hubaddr RW 0x0								prtaddr RW 0x0						

hcsplt10 Fields

Bit	Name	Description	Access	Reset
31	spltena	Split Enable (SpltEna) The application sets this field to indicate that this channel is enabled to perform split transactions. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
16	compsplt	Do Complete Split (CompSplt) The application sets this field to request the OTG host to perform a complete split transaction. **Value** — **Description** 0x0 — NOSPLIT 0x1 — SPLIT	RW	0x0

Bit	Name	Description	Access	Reset
15:14	xactpos	Transaction Position (XactPos) This field is used to determine whether to send all, first, middle, or last payloads with each OUT transaction. 2'b11: All. This is the entire data payload is of this transaction (which is less than or equal to 188 bytes). 2'b10: Begin. This is the first data payload of this transaction (which is larger than 188 bytes). 2'b00: Mid. This is the middle payload of this transaction (which is larger than 188 bytes). 2'b01: End. This is the last payload of this transaction (which is larger than 188 bytes). **Value** **Description** 0x0 MIDDLE 0x1 END 0x2 BEGIN 0x3 ALL	RW	0x0
13:7	hubaddr	Hub Address (HubAddr) This field holds the device address of the transaction translator's hub.	RW	0x0
6:0	prtaddr	Port Address (PrtAddr) This field is the port number of the recipient transaction translator.	RW	0x0

hcint10

Host Channel 10 Interrupt Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB00648

Module Instance	Base Address	Register Address
i_usbotg_1_hostgrp	0xFFB40400	0xFFB40648

Offset: 0x248

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		desc_lst_rollintr RW 0x0	xcs_xact_err RW 0x0	bnaintr RW 0x0	datatglerr RW 0x0	frmovrun RW 0x0	bblerr RW 0x0	xacterr RW 0x0	nyet RW 0x0	ack RW 0x0	nak RW 0x0	stall RW 0x0	ahberr RW 0x0	chhltd RW 0x0	xfercompl RW 0x0

hcint10 Fields

Bit	Name	Description	Access	Reset
13	desc_lst_rollintr	Descriptor rollover interrupt (DESC_LST_ROLLIntr) This bit is valid only when Scatter/Gather DMA mode is enabled. The core sets this bit when the corresponding channel's descriptor list rolls over. For non Scatter/Gather DMA mode, this bit is reserved. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
12	xcs_xact_err	Excessive Transaction Error (XCS_XACT_ERR) This bit is valid only when Scatter/Gather DMA mode is enabled. The core sets this bit when 3 consecutive transaction errors occurred on the USB bus. XCS_XACT_ERR will not be generated for Isochronous channels. For non Scatter/Gather DMA mode, this bit is reserved. **Value** **Description** 0x0 INACTIVE 0x1 ACVTIVE	RW	0x0
11	bnaintr	BNA (Buffer Not Available) Interrupt (BNAIntr) This bit is valid only when Scatter/Gather DMA mode is enabled. The core generates this interrupt when the descriptor accessed is not ready for the Core to process. BNA will not be generated for Isochronous channels. For non Scatter/Gather DMA mode, this bit is reserved. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
10	datatglerr	Data Toggle Error (DataTglErr).This bit can be set only by the core and the application should write 1 to clear it.In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
9	frmovrun	Frame Overrun (FrmOvrun).In Scatter/ Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the applica- tion should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
8	bblerr	Babble Error (BblErr) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core..This bit can be set only by the core and the applica- tion should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
7	xacterr	Transaction Error (XactErr) Indicates one of the following errors occurred on the USB. CRC check failure Timeout Bit stuff error False EOP In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the applica- tion should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
6	nyet	NYET Response Received Interrupt (NYET) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
5	ack	ACK Response Received/Transmitted Interrupt (ACK) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
4	nak	NAK Response Received Interrupt (NAK) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
3	stall	STALL Response Received Interrupt (STALL) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
2	ahberr	AHB Error (AHBErr) This is generated only in Internal DMA mode when there is an AHB error during AHB read/write. The application can read the corresponding channel's DMA address register to get the error address. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
1	chhltd	Channel Halted (ChHltd) In non Scatter/Gather DMA mode, it indicates the transfer completed abnormally either because of any USB transaction error or in response to disable request by the application or because of a completed transfer. in Scatter/gather DMA mode, this indicates that transfer completed due to any of the following . EOL being set in descriptor . AHB error . Excessive transaction errors . Babble . Stall **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
0	xfercompl	Transfer Completed (XferCompl) Transfer completed normally without any errors.This bit can be set only by the core and the application should write 1 to clear it. For Scatter/Gather DMA mode, it indicates that current descriptor processing got completed with IOC bit set in its descriptor. In non Scatter/Gather DMA mode, it indicates that Transfer completed normally without any errors. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

hcintmsk10

Host Channel 10 Interrupt Mask Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB0064C
i_usbotg_1_hostgrp	0xFFB40400	0xFFB4064C

Offset: 0x24C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		frm_lst_rollintrmsk RW 0x0	Reserved	bnaintrmsk RW 0x0	datatglerrmsk RW 0x0	frmovrunmsk RW 0x0	bblerrmsk RW 0x0	xacterrmsk RW 0x0	nyetmsk RW 0x0	ackmsk RW 0x0	nakmsk RW 0x0	stallmsk RW 0x0	ahberrmsk RW 0x0	chhltdmsk RW 0x0	xfercomplmsk RW 0x0

hcintmsk10 Fields

Bit	Name	Description	Access	Reset
13	frm_lst_rollintrmsk	Framelist rollover interrupt Mask register(FRM_LST_ROLLIntrMsk) This bit is valid only when Scatter/Gather DMA mode is enabled. **Value** — **Description** 0x0 — MASK 0x1 — NOMASK	RW	0x0

Bit	Name	Description	Access	Reset
11	bnaintrmsk	BNA (Buffer Not Available) Interrupt mask register (BNAIntrMsk) This bit is valid only when Scatter/Gather DMA mode is enabled. Value: 0x0 Description: MASK Value: 0x1 Description: NOMASK	RW	0x0
10	datatglerrmsk	Data Toggle Error Mask (DataTglErrMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
9	frmovrunmsk	Frame Overrun Mask (FrmOvrunMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
8	bblerrmsk	Babble Error Mask (BblErrMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
7	xacterrmsk	Transaction Error Mask (XactErrMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0

Bit	Name	Description	Access	Reset
6	nyetmsk	NYET Response Received Interrupt Mask (NyetMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
5	ackmsk	ACK Response Received/Transmitted Interrupt Mask (AckMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
4	nakmsk	NAK Response Received Interrupt Mask (NakMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
3	stallmsk	STALL Response Received Interrupt Mask (StallMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0

Bit	Name	Description	Access	Reset
2	ahberrmsk	AHB Error Mask (AHBErrMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn. **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0
1	chhltdmsk	Channel Halted Mask (ChHltdMsk) **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0
0	xfercomplmsk	Transfer Completed Mask (XferComplMsk) **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0

hctsiz10

Host Channel 10 Transfer Size Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB00650
i_usbotg_1_hostgrp	0xFFB40400	0xFFB40650

Offset: 0x250

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dopng RW 0x0	pid RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
xfersize RW 0x0															

hctsiz10 Fields

Bit	Name	Description	Access	Reset
31	dopng	Do Ping (DoPng) This bit is used only For OUT transfers. Setting this field to 1 directs the host to do PING protocol. Note: Do not Set this bit For IN transfers. If this bit is Set For for IN transfers it disables the channel. **Value** — **Description** 0x0 — NOPING 0x1 — PING	RW	0x0

Bit	Name	Description	Access	Reset
30:29	pid	PID (Pid) The application programs this field with the type of PID to use For the initial transaction. The host maintains this field For the rest of the transfer. 2'b00: DATA0 2'b01: DATA2 2'b10: DATA1 2'b11: MDATA (non-control)/SETUP (control) **Value** **Description** 0x0 DATA0 0x1 DATA2 0x2 DATA1 0x3 MDATA	RW	0x0
28:19	pktcnt	Packet Count (PktCnt) This field is programmed by the application with the expected number of packets to be transmitted (OUT) or received (IN). The host decrements this count on every successful transmission or reception of an OUT/IN packet. Once this count reaches zero, the application is interrupted to indicate normal completion. The width of this counter is specified as Width of Packet Counters	RW	0x0
18:0	xfersize	Transfer Size (XferSize) For an OUT, this field is the number of data bytes the host sends during the transfer. For an IN, this field is the buffer size that the application has Reserved For the transfer. The application is expected to program this field as an integer multiple of the maximum packet size For IN transactions (periodic and non-periodic). The width of this counter is specified as Width of Transfer Size Counters	RW	0x0

hcdma10

Host Channel 10 DMA Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB00654
i_usbotg_1_hostgrp	0xFFB40400	0xFFB40654

Offset: 0x254

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcdma10 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
hcdma10 RW 0x0															

hcdma10 Fields

Bit	Name	Description	Access	Reset
31:0	hcdma10	Buffer DMA Mode: [31:0] DMA Address (DMAAddr) This field holds the start address in the external memory from which the data for the endpoint must be fetched or to which it must be stored. This register is incremented on every AHB transac- tion. Scatter-Gather DMA (DescDMA) Mode: [31:9] (Non Isoc) Non-Isochronous: [31:N] (Isoc) Isochronous: This field holds the start address of the 512 bytes page. The first descriptor in the list should be located in this address. The first	RW	0x0

Bit	Name	Description	Access	Reset
		descriptor may be or may not be ready. The core starts processing the list from the CTD value. This field holds the address of the 2*(nTD+1) bytes of locations in which the isochronous descriptors are present where N is based on nTD as per Table below [31:N] Base Address [N-1:3] Offset [2:0] 000 HS ISOC nTD N 7 6 15 7 31 8 63 9 127 10 255 11 FS ISOC nTD N 1 4 3 5 7 6 15 7 31 8 63 9 [N-1:3] (Isoc): [8:3] (Non Isoc): Current Transfer Desc(CTD): Non Isochronous: This value is in terms of number of descriptors. The values can be from 0 to 63. 0 - 1 descriptor. 63 - 64 descriptors. This field indicates the current descriptor processed in the list. This field is updated both by application and the core. For example, if the application enables the channel after programming CTD=5, then the core will start processing the 6th descriptor. The address is obtained		

Bit	Name	Description	Access	Reset
		by adding a value of (8bytes*5=) 40(decimal) to DMAAddr. Isochronous: CTD for isochronous is based on the current frame/(micro)frame value. Need to be set to zero by application.Scatter-Gather DMA (DescDMA) Mode: [31:9] (Non Isoc) Non-Isochronous: [31:N] (Isoc) Isochronous: This field holds the start address of the 512 bytes page. The first descriptor in the list should be located in this address. The first descriptor may be or may not be ready. The core starts processing the list from the CTD value. This field holds the address of the 2*(nTD+1) bytes of locations in which the isochronous descriptors are present where N is based on nTD as per Table below [31:N] Base Address [N-1:3] Offset [2:0] 000 HS ISOC nTD N 7 6 15 7 31 8 63 9 127 10 255 11 FS ISOC nTD N 1 4 3 5 7 6 15 7 31 8 63 9 [N-1:3] (Isoc): [8:3] (Non Isoc): Current Transfer Desc(CTD):		

Bit	Name	Description	Access	Reset
		Non Isochronous: This value is in terms of number of descriptors. The values can be from 0 to 63. 0 - 1 descriptor. 63 - 64 descriptors. This field indicates the current descriptor processed in the list. This field is updated both by application and the core. For example, if the application enables the channel after programming CTD=5, then the core will start processing the 6th descriptor. The address is obtained by adding a value of (8bytes*5=) 40(decimal) to DMAAddr. Isochronous: CTD for isochronous is based on the current frame/(micro)frame value. Need to be set to zero by application.		

hcdmab10

Host Channel 10 DMA Buffer Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB0065C
i_usbotg_1_hostgrp	0xFFB40400	0xFFB4065C

Offset: 0x25C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcdmab10 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
hcdmab10 RW 0x0															

hcdmab10 Fields

Bit	Name	Description	Access	Reset
31:0	hcdmab10	Holds the current buffer address. This register is updated as and when the data transfer for the corresponding end point is in progress. This register is present only in Scatter/Gather DMA mode. Otherwise this field is reserved.	RW	0x0

hcchar11

Host Channel 11 Characteristics Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB00660
i_usbotg_1_hostgrp	0xFFB40400	0xFFB40660

Offset: 0x260

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
chena RW 0x0	chdis RW 0x0	oddfrm RW 0x0	devaddr RW 0x0							ec RW 0x0		eptype RW 0x0		lspddev RW 0x0	Reserved
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
epdir RW 0x0	epnum RW 0x0				mps RW 0x0										

hcchar11 Fields

Bit	Name	Description	Access	Reset
31	chena	Channel Enable (ChEna) When Scatter/Gather mode is enabled 1'b0: Indicates that the descriptor structure is not yet ready. 1'b1: Indicates that the descriptor structure and data buffer with data is setup and this channel can access the descriptor. When Scatter/Gather mode is disabled This field is set by the application and cleared by the OTG host. 1'b0: Channel disabled 1'b1: Channel enabled **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
30	chdis	Channel Disable (ChDis) The application sets this bit to stop transmitting/receiving data on a channel, even before the transfer For that channel is complete. The application must wait For the Channel Disabled interrupt before treating the channel as disabled. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
29	oddfrm	Odd Frame (OddFrm) This field is set (reset) by the application to indicate that the OTG host must perform a transfer in an odd (micro)frame. This field is applicable for only periodic (isochronous and interrupt) transactions. 1'b0: Even (micro)frame 1'b1: Odd (micro)frame	RW	0x0
28:22	devaddr	Device Address (DevAddr) This field selects the specific device serving as the data source or sink.	RW	0x0

Bit	Name	Description	Access	Reset
21:20	ec	Multi Count (MC) / Error Count (EC) When the Split Enable bit of the Host Channel-n Split Control register (HCSPLTn.SpltEna) is reset (1'b0), this field indicates to the host the number of transactions that must be executed per microframe For this periodic endpoint. For non periodic transfers, this field is used only in DMA mode, and specifies the number packets to be fetched For this channel before the internal DMA engine changes arbitration. 2'b00: Reserved This field yields undefined results. 2'b01: 1 transaction 2'b10: 2 transactions to be issued For this endpoint per microframe 2'b11: 3 transactions to be issued For this endpoint per microframe When HCSPLTn.SpltEna is Set (1'b1), this field indicates the number of immediate retries to be performed For a periodic split transactions on transaction errors. This field must be Set to at least 2'b01. **Value** **Description** 0x0 RESERVED 0x1 TRANSONE 0x2 TRANSTWO 0x3 TRANSTHREE	RW	0x0

Bit	Name	Description	Access	Reset
19:18	eptype	Endpoint Type (EPType) Indicates the transfer type selected. 2'b00: Control 2'b01: Isochronous 2'b10: Bulk 2'b11: Interrupt **Value** **Description** 0x0 CTRL 0x1 ISOC 0x2 BULK 0x3 INTERR	RW	0x0
17	lspddev	Low-Speed Device (LSpdDev) This field is Set by the application to indicate that this channel is communicating to a low-speed device. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
15	epdir	Endpoint Direction (EPDir) Indicates whether the transaction is IN or OUT. 1'b0: OUT 1'b1: IN **Value** **Description** 0x0 OUT 0x1 IN	RW	0x0

Bit	Name	Description	Access	Reset
14:11	epnum	Endpoint Number (EPNum) Indicates the endpoint number on the device serving as the data source or sink. Value: Description 0xa: ENDPT10 0xb: ENDPT11 0xc: ENDPT12 0xd: ENDPT13 0xe: ENDPT14 0xf: ENDPT15 0x0: ENDPT0 0x1: ENDPT1 0x2: ENDPT2 0x3: ENDPT3 0x4: ENDPT4 0x5: ENDPT5 0x6: ENDPT6 0x7: ENDPT7 0x8: ENDPT8 0x9: ENDPT9	RW	0x0
10:0	mps	Maximum Packet Size (MPS) Indicates the maximum packet size of the associated endpoint.	RW	0x0

hcsplt11

Host Channel 11 Split Control Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB00664
i_usbotg_1_hostgrp	0xFFB40400	0xFFB40664

Offset: 0x264

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
spltena RW 0x0	Reserved														compsplt RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
xactpos RW 0x0	hubaddr RW 0x0								prtaddr RW 0x0						

hcsplt11 Fields

Bit	Name	Description	Access	Reset
31	spltena	Split Enable (SpltEna) The application sets this field to indicate that this channel is enabled to perform split transactions. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
16	compsplt	Do Complete Split (CompSplt) The application sets this field to request the OTG host to perform a complete split transaction. **Value** — **Description** 0x0 — NOSPLIT 0x1 — SPLIT	RW	0x0

Bit	Name	Description	Access	Reset
15:14	xactpos	Transaction Position (XactPos) This field is used to determine whether to send all, first, middle, or last payloads with each OUT transaction. 2'b11: All. This is the entire data payload is of this transaction (which is less than or equal to 188 bytes). 2'b10: Begin. This is the first data payload of this transaction (which is larger than 188 bytes). 2'b00: Mid. This is the middle payload of this transaction (which is larger than 188 bytes). 2'b01: End. This is the last payload of this transaction (which is larger than 188 bytes). **Value** **Description** 0x0 MIDDLE 0x1 END 0x2 BEGIN 0x3 ALL	RW	0x0
13:7	hubaddr	Hub Address (HubAddr) This field holds the device address of the transaction translator's hub.	RW	0x0
6:0	prtaddr	Port Address (PrtAddr) This field is the port number of the recipient transaction translator.	RW	0x0

hcint11

Host Channel 11 Interrupt Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB00668

Module Instance	Base Address	Register Address
i_usbotg_1_hostgrp	0xFFB40400	0xFFB40668

Offset: 0x268

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		desc_lst_rollintr RW 0x0	xcs_xact_err RW 0x0	bnaintr RW 0x0	datatglerr RW 0x0	frmovrun RW 0x0	bblerr RW 0x0	xacterr RW 0x0	nyet RW 0x0	ack RW 0x0	nak RW 0x0	stall RW 0x0	ahberr RW 0x0	chhltd RW 0x0	xfercompl RW 0x0

hcint11 Fields

Bit	Name	Description	Access	Reset
13	desc_lst_rollintr	Descriptor rollover interrupt (DESC_LST_ROLLIntr) This bit is valid only when Scatter/Gather DMA mode is enabled. The core sets this bit when the corresponding channel's descriptor list rolls over. For non Scatter/Gather DMA mode, this bit is reserved. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
12	xcs_xact_err	Excessive Transaction Error (XCS_XACT_ERR) This bit is valid only when Scatter/Gather DMA mode is enabled. The core sets this bit when 3 consecutive transaction errors occurred on the USB bus. XCS_XACT_ERR will not be generated for Isochronous channels. For non Scatter/Gather DMA mode, this bit is reserved. **Value** **Description** 0x0 INACTIVE 0x1 ACVTIVE	RW	0x0
11	bnaintr	BNA (Buffer Not Available) Interrupt (BNAIntr) This bit is valid only when Scatter/Gather DMA mode is enabled. The core generates this interrupt when the descriptor accessed is not ready for the Core to process. BNA will not be generated for Isochronous channels. For non Scatter/Gather DMA mode, this bit is reserved. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
10	datatglerr	Data Toggle Error (DataTglErr).This bit can be set only by the core and the application should write 1 to clear it.In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
9	frmovrun	Frame Overrun (FrmOvrun).In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
8	bblerr	Babble Error (BblErr) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core..This bit can be set only by the core and the application should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
7	xacterr	Transaction Error (XactErr) Indicates one of the following errors occurred on the USB. CRC check failure Timeout Bit stuff error False EOP In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
6	nyet	NYET Response Received Interrupt (NYET) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
5	ack	ACK Response Received/Transmitted Interrupt (ACK) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
4	nak	NAK Response Received Interrupt (NAK) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
3	stall	STALL Response Received Interrupt (STALL) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
2	ahberr	AHB Error (AHBErr) This is generated only in Internal DMA mode when there is an AHB error during AHB read/write. The application can read the corresponding channel's DMA address register to get the error address. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
1	chhltd	Channel Halted (ChHltd) In non Scatter/Gather DMA mode, it indicates the transfer completed abnormally either because of any USB transaction error or in response to disable request by the application or because of a completed transfer. in Scatter/gather DMA mode, this indicates that transfer completed due to any of the following . EOL being set in descriptor . AHB error . Excessive transaction errors . Babble . Stall **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
0	xfercompl	Transfer Completed (XferCompl) Transfer completed normally without any errors.This bit can be set only by the core and the application should write 1 to clear it. For Scatter/Gather DMA mode, it indicates that current descriptor processing got completed with IOC bit set in its descriptor. In non Scatter/Gather DMA mode, it indicates that Transfer completed normally without any errors. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

hcintmsk11

Host Channel 11 Interrupt Mask Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB0066C
i_usbotg_1_hostgrp	0xFFB40400	0xFFB4066C

Offset: 0x26C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		frm_lst_rollintrmsk RW 0x0	Reserved	bnaintrmsk RW 0x0	datatglerrmsk RW 0x0	frmovrunmsk RW 0x0	bblerrmsk RW 0x0	xacterrmsk RW 0x0	nyetmsk RW 0x0	ackmsk RW 0x0	nakmsk RW 0x0	stallmsk RW 0x0	ahberrmsk RW 0x0	chhltdmsk RW 0x0	xfercomplmsk RW 0x0

hcintmsk11 Fields

Bit	Name	Description	Access	Reset
13	frm_lst_rollintrmsk	Framelist rollover interrupt Mask register(FRM_LST_ROLLIntrMsk) This bit is valid only when Scatter/Gather DMA mode is enabled. **Value** — **Description** 0x0 — MASK 0x1 — NOMASK	RW	0x0

Bit	Name	Description	Access	Reset
11	bnaintrmsk	BNA (Buffer Not Available) Interrupt mask register (BNAIntrMsk) This bit is valid only when Scatter/Gather DMA mode is enabled. **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0
10	datatglerrmsk	Data Toggle Error Mask (DataTglErrMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
9	frmovrunmsk	Frame Overrun Mask (FrmOvrunMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
8	bblerrmsk	Babble Error Mask (BblErrMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
7	xacterrmsk	Transaction Error Mask (XactErrMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0

Bit	Name	Description	Access	Reset
6	nyetmsk	NYET Response Received Interrupt Mask (NyetMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
5	ackmsk	ACK Response Received/Transmitted Interrupt Mask (AckMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
4	nakmsk	NAK Response Received Interrupt Mask (NakMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
3	stallmsk	STALL Response Received Interrupt Mask (StallMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0

Bit	Name	Description	Access	Reset
2	ahberrmsk	AHB Error Mask (AHBErrMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn. **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0
1	chhltdmsk	Channel Halted Mask (ChHltdMsk) **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0
0	xfercomplmsk	Transfer Completed Mask (XferComplMsk) **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0

hctsiz11

Host Channel 11 Transfer Size Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB00670
i_usbotg_1_hostgrp	0xFFB40400	0xFFB40670

Offset: 0x270

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dopng RW 0x0	pid RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
xfersize RW 0x0															

hctsiz11 Fields

Bit	Name	Description	Access	Reset
31	dopng	Do Ping (DoPng) This bit is used only For OUT transfers. Setting this field to 1 directs the host to do PING protocol. Note: Do not Set this bit For IN transfers. If this bit is Set For for IN transfers it disables the channel. **Value** — **Description** 0x0 — NOPING 0x1 — PING	RW	0x0

Bit	Name	Description	Access	Reset
30:29	pid	PID (Pid) The application programs this field with the type of PID to use For the initial transaction. The host maintains this field For the rest of the transfer. 2'b00: DATA0 2'b01: DATA2 2'b10: DATA1 2'b11: MDATA (non-control)/SETUP (control) **Value** **Description** 0x0 DATA0 0x1 DATA2 0x2 DATA1 0x3 MDATA	RW	0x0
28:19	pktcnt	Packet Count (PktCnt) This field is programmed by the application with the expected number of packets to be transmitted (OUT) or received (IN). The host decrements this count on every successful transmission or reception of an OUT/IN packet. Once this count reaches zero, the application is interrupted to indicate normal completion. The width of this counter is specified as Width of Packet Counters	RW	0x0
18:0	xfersize	Transfer Size (XferSize) For an OUT, this field is the number of data bytes the host sends during the transfer. For an IN, this field is the buffer size that the application has Reserved For the transfer. The application is expected to program this field as an integer multiple of the maximum packet size For IN transactions (periodic and non-periodic). The width of this counter is specified as Width of Transfer Size Counters	RW	0x0

hcdma11

Host Channel 11 DMA Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB00674
i_usbotg_1_hostgrp	0xFFB40400	0xFFB40674

Offset: 0x274

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcdma11 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
hcdma11 RW 0x0															

hcdma11 Fields

Bit	Name	Description	Access	Reset
31:0	hcdma11	Buffer DMA Mode: [31:0] DMA Address (DMAAddr) This field holds the start address in the external memory from which the data for the endpoint must be fetched or to which it must be stored. This register is incremented on every AHB transac- tion. Scatter-Gather DMA (DescDMA) Mode: [31:9] (Non Isoc) Non-Isochronous: [31:N] (Isoc) Isochronous: This field holds the start address of the 512 bytes page. The first descriptor in the list should be located in this address. The first	RW	0x0

Bit	Name	Description	Access	Reset
		descriptor may be or may not be ready. The core starts processing the list from the CTD value. This field holds the address of the 2*(nTD+1) bytes of locations in which the isochronous descriptors are present where N is based on nTD as per Table below [31:N] Base Address [N-1:3] Offset [2:0] 000 HS ISOC nTD N 7 6 15 7 31 8 63 9 127 10 255 11 FS ISOC nTD N 1 4 3 5 7 6 15 7 31 8 63 9 [N-1:3] (Isoc): [8:3] (Non Isoc): Current Transfer Desc(CTD): Non Isochronous: This value is in terms of number of descriptors. The values can be from 0 to 63. 0 - 1 descriptor. 63 - 64 descriptors. This field indicates the current descriptor processed in the list. This field is updated both by application and the core. For example, if the application enables the channel after programming CTD=5, then the core will start processing the 6th descriptor. The address is obtained		

Bit	Name	Description	Access	Reset
		by adding a value of (8bytes*5=) 40(decimal) to DMAAddr. Isochronous: CTD for isochronous is based on the current frame/(micro)frame value. Need to be set to zero by application.Scatter-Gather DMA (DescDMA) Mode: [31:9] (Non Isoc) Non-Isochronous: [31:N] (Isoc) Isochronous: This field holds the start address of the 512 bytes page. The first descriptor in the list should be located in this address. The first descriptor may be or may not be ready. The core starts processing the list from the CTD value. This field holds the address of the 2*(nTD+1) bytes of locations in which the isochronous descriptors are present where N is based on nTD as per Table below [31:N] Base Address [N-1:3] Offset [2:0] 000 HS ISOC nTD N 7 6 15 7 31 8 63 9 127 10 255 11 FS ISOC nTD N 1 4 3 5 7 6 15 7 31 8 63 9 [N-1:3] (Isoc): [8:3] (Non Isoc): Current Transfer Desc(CTD):		

Bit	Name	Description	Access	Reset
		Non Isochronous: This value is in terms of number of descriptors. The values can be from 0 to 63. 0 - 1 descriptor. 63 - 64 descriptors. This field indicates the current descriptor processed in the list. This field is updated both by application and the core. For example, if the application enables the channel after programming CTD=5, then the core will start processing the 6th descriptor. The address is obtained by adding a value of (8bytes*5=) 40(decimal) to DMAAddr. Isochronous: CTD for isochronous is based on the current frame/(micro)frame value. Need to be set to zero by application.		

hcdmab11

Host Channel 11 DMA Buffer Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB0067C
i_usbotg_1_hostgrp	0xFFB40400	0xFFB4067C

Offset: 0x27C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcdmab11 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
hcdmab11 RW 0x0															

hcdmab11 Fields

Bit	Name	Description	Access	Reset
31:0	hcdmab11	Holds the current buffer address. This register is updated as and when the data transfer for the corresponding end point is in progress. This register is present only in Scatter/Gather DMA mode. Otherwise this field is reserved.	RW	0x0

hcchar12

Host Channel 12 Characteristics Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB00680
i_usbotg_1_hostgrp	0xFFB40400	0xFFB40680

Offset: 0x280

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
chena RW 0x0	chdis RW 0x0	oddfrm RW 0x0	devaddr RW 0x0							ec RW 0x0		eptype RW 0x0		lspddev RW 0x0	Reserved
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
epdir RW 0x0	epnum RW 0x0				mps RW 0x0										

hcchar12 Fields

Bit	Name	Description	Access	Reset
31	chena	Channel Enable (ChEna) When Scatter/Gather mode is enabled 1'b0: Indicates that the descriptor structure is not yet ready. 1'b1: Indicates that the descriptor structure and data buffer with data is setup and this channel can access the descriptor. When Scatter/Gather mode is disabled This field is set by the application and cleared by the OTG host. 1'b0: Channel disabled 1'b1: Channel enabled **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
30	chdis	Channel Disable (ChDis) The application sets this bit to stop transmitting/receiving data on a channel, even before the transfer For that channel is complete. The application must wait For the Channel Disabled interrupt before treating the channel as disabled. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
29	oddfrm	Odd Frame (OddFrm) This field is set (reset) by the application to indicate that the OTG host must perform a transfer in an odd (micro)frame. This field is applicable for only periodic (isochronous and interrupt) transactions. 1'b0: Even (micro)frame 1'b1: Odd (micro)frame	RW	0x0
28:22	devaddr	Device Address (DevAddr) This field selects the specific device serving as the data source or sink.	RW	0x0

Bit	Name	Description	Access	Reset
21:20	ec	Multi Count (MC) / Error Count (EC) When the Split Enable bit of the Host Channel-n Split Control register (HCSPLTn.SpltEna) is reset (1'b0), this field indicates to the host the number of transactions that must be executed per microframe For this periodic endpoint. For non periodic transfers, this field is used only in DMA mode, and specifies the number packets to be fetched For this channel before the internal DMA engine changes arbitration. 2'b00: Reserved This field yields undefined results. 2'b01: 1 transaction 2'b10: 2 transactions to be issued For this endpoint per microframe 2'b11: 3 transactions to be issued For this endpoint per microframe When HCSPLTn.SpltEna is Set (1'b1), this field indicates the number of immediate retries to be performed For a periodic split transactions on transaction errors. This field must be Set to at least 2'b01. **Value** **Description** 0x0 RESERVED 0x1 TRANSONE 0x2 TRANSTWO 0x3 TRANSTHREE	RW	0x0

Bit	Name	Description	Access	Reset
19:18	eptype	Endpoint Type (EPType) Indicates the transfer type selected. 2'b00: Control 2'b01: Isochronous 2'b10: Bulk 2'b11: Interrupt **Value** — **Description** 0x0 — CTRL 0x1 — ISOC 0x2 — BULK 0x3 — INTERR	RW	0x0
17	lspddev	Low-Speed Device (LSpdDev) This field is Set by the application to indicate that this channel is communicating to a low-speed device. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
15	epdir	Endpoint Direction (EPDir) Indicates whether the transaction is IN or OUT. 1'b0: OUT 1'b1: IN **Value** — **Description** 0x0 — OUT 0x1 — IN	RW	0x0

Bit	Name	Description	Access	Reset
14:11	epnum	Endpoint Number (EPNum) Indicates the endpoint number on the device serving as the data source or sink. Value — Description 0xa — ENDPT10 0xb — ENDPT11 0xc — ENDPT12 0xd — ENDPT13 0xe — ENDPT14 0xf — ENDPT15 0x0 — ENDPT0 0x1 — ENDPT1 0x2 — ENDPT2 0x3 — ENDPT3 0x4 — ENDPT4 0x5 — ENDPT5 0x6 — ENDPT6 0x7 — ENDPT7 0x8 — ENDPT8 0x9 — ENDPT9	RW	0x0
10:0	mps	Maximum Packet Size (MPS) Indicates the maximum packet size of the associated endpoint.	RW	0x0

hcsplt12

Host Channel 12 Split Control Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB00684
i_usbotg_1_hostgrp	0xFFB40400	0xFFB40684

Offset: 0x284

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
spltena RW 0x0	Reserved														compsplt RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
xactpos RW 0x0	hubaddr RW 0x0								prtaddr RW 0x0						

hcsplt12 Fields

Bit	Name	Description	Access	Reset
31	spltena	Split Enable (SpltEna) The application sets this field to indicate that this channel is enabled to perform split transactions. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
16	compsplt	Do Complete Split (CompSplt) The application sets this field to request the OTG host to perform a complete split transaction. **Value** — **Description** 0x0 — NOSPLIT 0x1 — SPLIT	RW	0x0

Bit	Name	Description	Access	Reset
15:14	xactpos	Transaction Position (XactPos) This field is used to determine whether to send all, first, middle, or last payloads with each OUT transaction. 2'b11: All. This is the entire data payload is of this transaction (which is less than or equal to 188 bytes). 2'b10: Begin. This is the first data payload of this transaction (which is larger than 188 bytes). 2'b00: Mid. This is the middle payload of this transaction (which is larger than 188 bytes). 2'b01: End. This is the last payload of this transaction (which is larger than 188 bytes). **Value** **Description** 0x0 MIDDLE 0x1 END 0x2 BEGIN 0x3 ALL	RW	0x0
13:7	hubaddr	Hub Address (HubAddr) This field holds the device address of the transaction translator's hub.	RW	0x0
6:0	prtaddr	Port Address (PrtAddr) This field is the port number of the recipient transaction translator.	RW	0x0

hcint12

Host Channel 12 Interrupt Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB00688

Module Instance	Base Address	Register Address
i_usbotg_1_hostgrp	0xFFB40400	0xFFB40688

Offset: 0x288

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		desc_lst_rollintr RW 0x0	xcs_xact_err RW 0x0	bnaintr RW 0x0	datatglerr RW 0x0	frmovrun RW 0x0	bblerr RW 0x0	xacterr RW 0x0	nyet RW 0x0	ack RW 0x0	nak RW 0x0	stall RW 0x0	ahberr RW 0x0	chhltd RW 0x0	xfercompl RW 0x0

hcint12 Fields

Bit	Name	Description	Access	Reset
13	desc_lst_rollintr	Descriptor rollover interrupt (DESC_LST_ROLLIntr) This bit is valid only when Scatter/Gather DMA mode is enabled. The core sets this bit when the corresponding channel's descriptor list rolls over. For non Scatter/Gather DMA mode, this bit is reserved. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
12	xcs_xact_err	Excessive Transaction Error (XCS_XACT_ERR) This bit is valid only when Scatter/Gather DMA mode is enabled. The core sets this bit when 3 consecutive transaction errors occurred on the USB bus. XCS_XACT_ERR will not be generated for Isochronous channels. For non Scatter/Gather DMA mode, this bit is reserved. **Value** **Description** 0x0 INACTIVE 0x1 ACVTIVE	RW	0x0
11	bnaintr	BNA (Buffer Not Available) Interrupt (BNAIntr) This bit is valid only when Scatter/Gather DMA mode is enabled. The core generates this interrupt when the descriptor accessed is not ready for the Core to process. BNA will not be generated for Isochronous channels. For non Scatter/Gather DMA mode, this bit is reserved. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
10	datatglerr	Data Toggle Error (DataTglErr).This bit can be set only by the core and the application should write 1 to clear it.In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
9	frmovrun	Frame Overrun (FrmOvrun).In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
8	bblerr	Babble Error (BblErr) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core..This bit can be set only by the core and the application should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
7	xacterr	Transaction Error (XactErr) Indicates one of the following errors occurred on the USB. CRC check failure Timeout Bit stuff error False EOP In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
6	nyet	NYET Response Received Interrupt (NYET) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
5	ack	ACK Response Received/Transmitted Interrupt (ACK) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
4	nak	NAK Response Received Interrupt (NAK) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
3	stall	STALL Response Received Interrupt (STALL) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
2	ahberr	AHB Error (AHBErr) This is generated only in Internal DMA mode when there is an AHB error during AHB read/write. The application can read the corresponding channel's DMA address register to get the error address. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
1	chhltd	Channel Halted (ChHltd) In non Scatter/Gather DMA mode, it indicates the transfer completed abnormally either because of any USB transaction error or in response to disable request by the application or because of a completed transfer. in Scatter/gather DMA mode, this indicates that transfer completed due to any of the following . EOL being set in descriptor . AHB error . Excessive transaction errors . Babble . Stall **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
0	xfercompl	Transfer Completed (XferCompl) Transfer completed normally without any errors.This bit can be set only by the core and the application should write 1 to clear it. For Scatter/Gather DMA mode, it indicates that current descriptor processing got completed with IOC bit set in its descriptor. In non Scatter/Gather DMA mode, it indicates that Transfer completed normally without any errors. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

hcintmsk12

Host Channel 12 Interrupt Mask Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB0068C
i_usbotg_1_hostgrp	0xFFB40400	0xFFB4068C

Offset: 0x28C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		frm_lst_rollintrmsk RW 0x0	Reserved	bnaintrmsk RW 0x0	datatglerrmsk RW 0x0	frmovrunmsk RW 0x0	bblerrmsk RW 0x0	xacterrmsk RW 0x0	nyetmsk RW 0x0	ackmsk RW 0x0	nakmsk RW 0x0	stallmsk RW 0x0	ahberrmsk RW 0x0	chhltdmsk RW 0x0	xfercomplmsk RW 0x0

hcintmsk12 Fields

Bit	Name	Description	Access	Reset
13	frm_lst_rollintrmsk	Framelist rollover interrupt Mask register(FRM_LST_ROLLIntrMsk) This bit is valid only when Scatter/Gather DMA mode is enabled. **Value** — **Description** 0x0 — MASK 0x1 — NOMASK	RW	0x0

Bit	Name	Description	Access	Reset
11	bnaintrmsk	BNA (Buffer Not Available) Interrupt mask register (BNAIntrMsk) This bit is valid only when Scatter/Gather DMA mode is enabled. Value: 0x0 Description: MASK Value: 0x1 Description: NOMASK	RW	0x0
10	datatglerrmsk	Data Toggle Error Mask (DataTglErrMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
9	frmovrunmsk	Frame Overrun Mask (FrmOvrunMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
8	bblerrmsk	Babble Error Mask (BblErrMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
7	xacterrmsk	Transaction Error Mask (XactErrMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0

Bit	Name	Description	Access	Reset
6	nyetmsk	NYET Response Received Interrupt Mask (NyetMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
5	ackmsk	ACK Response Received/Transmitted Interrupt Mask (AckMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
4	nakmsk	NAK Response Received Interrupt Mask (NakMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
3	stallmsk	STALL Response Received Interrupt Mask (StallMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0

Bit	Name	Description	Access	Reset
2	ahberrmsk	AHB Error Mask (AHBErrMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn. **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0
1	chhltdmsk	Channel Halted Mask (ChHltdMsk) **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0
0	xfercomplmsk	Transfer Completed Mask (XferComplMsk) **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0

hctsiz12

Host Channel 12 Transfer Size Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB00690
i_usbotg_1_hostgrp	0xFFB40400	0xFFB40690

Offset: 0x290

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dopng RW 0x0	pid RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
xfersize RW 0x0															

hctsiz12 Fields

Bit	Name	Description	Access	Reset
31	dopng	Do Ping (DoPng) This bit is used only For OUT transfers. Setting this field to 1 directs the host to do PING protocol. Note: Do not Set this bit For IN transfers. If this bit is Set For for IN transfers it disables the channel. **Value** — **Description** 0x0 — NOPING 0x1 — PING	RW	0x0

Bit	Name	Description	Access	Reset
30:29	pid	PID (Pid) The application programs this field with the type of PID to use For the initial transaction. The host maintains this field For the rest of the transfer. 2'b00: DATA0 2'b01: DATA2 2'b10: DATA1 2'b11: MDATA (non-control)/SETUP (control) **Value** **Description** 0x0 DATA0 0x1 DATA2 0x2 DATA1 0x3 MDATA	RW	0x0
28:19	pktcnt	Packet Count (PktCnt) This field is programmed by the application with the expected number of packets to be transmitted (OUT) or received (IN). The host decrements this count on every successful transmission or reception of an OUT/IN packet. Once this count reaches zero, the application is interrupted to indicate normal completion. The width of this counter is specified as Width of Packet Counters	RW	0x0
18:0	xfersize	Transfer Size (XferSize) For an OUT, this field is the number of data bytes the host sends during the transfer. For an IN, this field is the buffer size that the application has Reserved For the transfer. The application is expected to program this field as an integer multiple of the maximum packet size For IN transactions (periodic and non-periodic). The width of this counter is specified as Width of Transfer Size Counters	RW	0x0

hcdma12

Host Channel 12 DMA Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB00694
i_usbotg_1_hostgrp	0xFFB40400	0xFFB40694

Offset: 0x294

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcdma12 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
hcdma12 RW 0x0															

hcdma12 Fields

Bit	Name	Description	Access	Reset
31:0	hcdma12	Buffer DMA Mode: [31:0] DMA Address (DMAAddr) This field holds the start address in the external memory from which the data for the endpoint must be fetched or to which it must be stored. This register is incremented on every AHB transac- tion. Scatter-Gather DMA (DescDMA) Mode: [31:9] (Non Isoc) Non-Isochronous: [31:N] (Isoc) Isochronous: This field holds the start address of the 512 bytes page. The first descriptor in the list should be located in this address. The first	RW	0x0

Bit	Name	Description	Access	Reset
		descriptor may be or may not be ready. The core starts processing the list from the CTD value. This field holds the address of the 2*(nTD+1) bytes of locations in which the isochronous descriptors are present where N is based on nTD as per Table below [31:N] Base Address [N-1:3] Offset [2:0] 000 HS ISOC nTD N 7 6 15 7 31 8 63 9 127 10 255 11 FS ISOC nTD N 1 4 3 5 7 6 15 7 31 8 63 9 [N-1:3] (Isoc): [8:3] (Non Isoc): Current Transfer Desc(CTD): Non Isochronous: This value is in terms of number of descriptors. The values can be from 0 to 63. 0 - 1 descriptor. 63 - 64 descriptors. This field indicates the current descriptor processed in the list. This field is updated both by application and the core. For example, if the application enables the channel after programming CTD=5, then the core will start processing the 6th descriptor. The address is obtained		

Bit	Name	Description	Access	Reset
		by adding a value of (8bytes*5=) 40(decimal) to DMAAddr. Isochronous: CTD for isochronous is based on the current frame/(micro)frame value. Need to be set to zero by application.Scatter-Gather DMA (DescDMA) Mode: [31:9] (Non Isoc) Non-Isochronous: [31:N] (Isoc) Isochronous: This field holds the start address of the 512 bytes page. The first descriptor in the list should be located in this address. The first descriptor may be or may not be ready. The core starts processing the list from the CTD value. This field holds the address of the 2*(nTD+1) bytes of locations in which the isochronous descriptors are present where N is based on nTD as per Table below [31:N] Base Address [N-1:3] Offset [2:0] 000 HS ISOC nTD N 7 6 15 7 31 8 63 9 127 10 255 11 FS ISOC nTD N 1 4 3 5 7 6 15 7 31 8 63 9 [N-1:3] (Isoc): [8:3] (Non Isoc): Current Transfer Desc(CTD):		

Bit	Name	Description	Access	Reset
		Non Isochronous: This value is in terms of number of descriptors. The values can be from 0 to 63. 0 - 1 descriptor. 63 - 64 descriptors. This field indicates the current descriptor processed in the list. This field is updated both by application and the core. For example, if the application enables the channel after programming CTD=5, then the core will start processing the 6th descriptor. The address is obtained by adding a value of (8bytes*5=) 40(decimal) to DMAAddr. Isochronous: CTD for isochronous is based on the current frame/(micro)frame value. Need to be set to zero by application.		

hcdmab12

Host Channel 12 DMA Buffer Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB0069C
i_usbotg_1_hostgrp	0xFFB40400	0xFFB4069C

Offset: 0x29C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcdmab12 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
hcdmab12 RW 0x0															

hcdmab12 Fields

Bit	Name	Description	Access	Reset
31:0	hcdmab12	Holds the current buffer address. This register is updated as and when the data transfer for the corresponding end point is in progress. This register is present only in Scatter/Gather DMA mode. Otherwise this field is reserved.	RW	0x0

hcchar13

Host Channel 13 Characteristics Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB006A0
i_usbotg_1_hostgrp	0xFFB40400	0xFFB406A0

Offset: 0x2A0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
chena RW 0x0	chdis RW 0x0	oddfrm RW 0x0	devaddr RW 0x0							ec RW 0x0		eptype RW 0x0		lspddev RW 0x0	Reserved
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
epdir RW 0x0	epnum RW 0x0				mps RW 0x0										

hcchar13 Fields

Bit	Name	Description	Access	Reset
31	chena	Channel Enable (ChEna) When Scatter/Gather mode is enabled 1'b0: Indicates that the descriptor structure is not yet ready. 1'b1: Indicates that the descriptor structure and data buffer with data is setup and this channel can access the descriptor. When Scatter/Gather mode is disabled This field is set by the application and cleared by the OTG host. 1'b0: Channel disabled 1'b1: Channel enabled **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
30	chdis	Channel Disable (ChDis) The application sets this bit to stop transmitting/receiving data on a channel, even before the transfer For that channel is complete. The application must wait For the Channel Disabled interrupt before treating the channel as disabled. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
29	oddfrm	Odd Frame (OddFrm) This field is set (reset) by the application to indicate that the OTG host must perform a transfer in an odd (micro)frame. This field is applicable for only periodic (isochronous and interrupt) transactions. 1'b0: Even (micro)frame 1'b1: Odd (micro)frame	RW	0x0
28:22	devaddr	Device Address (DevAddr) This field selects the specific device serving as the data source or sink.	RW	0x0

Bit	Name	Description	Access	Reset
21:20	ec	Multi Count (MC) / Error Count (EC) When the Split Enable bit of the Host Channel-n Split Control register (HCSPLTn.SpltEna) is reset (1'b0), this field indicates to the host the number of transactions that must be executed per microframe For this periodic endpoint. For non periodic transfers, this field is used only in DMA mode, and specifies the number packets to be fetched For this channel before the internal DMA engine changes arbitration. 2'b00: Reserved This field yields undefined results. 2'b01: 1 transaction 2'b10: 2 transactions to be issued For this endpoint per microframe 2'b11: 3 transactions to be issued For this endpoint per microframe When HCSPLTn.SpltEna is Set (1'b1), this field indicates the number of immediate retries to be performed For a periodic split transactions on transaction errors. This field must be Set to at least 2'b01. **Value** **Description** 0x0 RESERVED 0x1 TRANSONE 0x2 TRANSTWO 0x3 TRANSTHREE	RW	0x0

Bit	Name	Description	Access	Reset
19:18	eptype	Endpoint Type (EPType) Indicates the transfer type selected. 2'b00: Control 2'b01: Isochronous 2'b10: Bulk 2'b11: Interrupt **Value** / **Description** 0x0 — CTRL 0x1 — ISOC 0x2 — BULK 0x3 — INTERR	RW	0x0
17	lspddev	Low-Speed Device (LSpdDev) This field is Set by the application to indicate that this channel is communicating to a low-speed device. **Value** / **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
15	epdir	Endpoint Direction (EPDir) Indicates whether the transaction is IN or OUT. 1'b0: OUT 1'b1: IN **Value** / **Description** 0x0 — OUT 0x1 — IN	RW	0x0

Bit	Name	Description	Access	Reset
14:11	epnum	Endpoint Number (EPNum) Indicates the endpoint number on the device serving as the data source or sink. Value — Description 0xa — ENDPT10 0xb — ENDPT11 0xc — ENDPT12 0xd — ENDPT13 0xe — ENDPT14 0xf — ENDPT15 0x0 — ENDPT0 0x1 — ENDPT1 0x2 — ENDPT2 0x3 — ENDPT3 0x4 — ENDPT4 0x5 — ENDPT5 0x6 — ENDPT6 0x7 — ENDPT7 0x8 — ENDPT8 0x9 — ENDPT9	RW	0x0
10:0	mps	Maximum Packet Size (MPS) Indicates the maximum packet size of the associated endpoint.	RW	0x0

hcsplt13

Host Channel 13 Split Control Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB006A4
i_usbotg_1_hostgrp	0xFFB40400	0xFFB406A4

Offset: 0x2A4

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
spltena RW 0x0	Reserved														compsplt RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
xactpos RW 0x0	hubaddr RW 0x0								prtaddr RW 0x0						

hcsplt13 Fields

Bit	Name	Description	Access	Reset
31	spltena	Split Enable (SpltEna) The application sets this field to indicate that this channel is enabled to perform split transactions. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
16	compsplt	Do Complete Split (CompSplt) The application sets this field to request the OTG host to perform a complete split transaction. **Value** — **Description** 0x0 — NOSPLIT 0x1 — SPLIT	RW	0x0

Bit	Name	Description	Access	Reset
15:14	xactpos	Transaction Position (XactPos) This field is used to determine whether to send all, first, middle, or last payloads with each OUT transaction. 2'b11: All. This is the entire data payload is of this transaction (which is less than or equal to 188 bytes). 2'b10: Begin. This is the first data payload of this transaction (which is larger than 188 bytes). 2'b00: Mid. This is the middle payload of this transaction (which is larger than 188 bytes). 2'b01: End. This is the last payload of this transaction (which is larger than 188 bytes). **Value** **Description** 0x0 MIDDLE 0x1 END 0x2 BEGIN 0x3 ALL	RW	0x0
13:7	hubaddr	Hub Address (HubAddr) This field holds the device address of the transaction translator's hub.	RW	0x0
6:0	prtaddr	Port Address (PrtAddr) This field is the port number of the recipient transaction translator.	RW	0x0

hcint13

Host Channel 13 Interrupt Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB006A8

Module Instance	Base Address	Register Address
i_usbotg_1_hostgrp	0xFFB40400	0xFFB406A8

Offset: 0x2A8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		desc_lst_rollintr RW 0x0	xcs_xact_err RW 0x0	bnaintr RW 0x0	datatglerr RW 0x0	frmovrun RW 0x0	bblerr RW 0x0	xacterr RW 0x0	nyet RW 0x0	ack RW 0x0	nak RW 0x0	stall RW 0x0	ahberr RW 0x0	chhltd RW 0x0	xfercompl RW 0x0

hcint13 Fields

Bit	Name	Description	Access	Reset
13	desc_lst_rollintr	Descriptor rollover interrupt (DESC_LST_ROLLIntr) This bit is valid only when Scatter/Gather DMA mode is enabled. The core sets this bit when the corresponding channel's descriptor list rolls over. For non Scatter/Gather DMA mode, this bit is reserved. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
12	xcs_xact_err	Excessive Transaction Error (XCS_XACT_ERR) This bit is valid only when Scatter/Gather DMA mode is enabled. The core sets this bit when 3 consecutive transaction errors occurred on the USB bus. XCS_XACT_ERR will not be generated for Isochronous channels. For non Scatter/Gather DMA mode, this bit is reserved. **Value** **Description** 0x0 INACTIVE 0x1 ACVTIVE	RW	0x0
11	bnaintr	BNA (Buffer Not Available) Interrupt (BNAIntr) This bit is valid only when Scatter/Gather DMA mode is enabled. The core generates this interrupt when the descriptor accessed is not ready for the Core to process. BNA will not be generated for Isochronous channels. For non Scatter/Gather DMA mode, this bit is reserved. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
10	datatglerr	Data Toggle Error (DataTglErr).This bit can be set only by the core and the application should write 1 to clear it.In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
9	frmovrun	Frame Overrun (FrmOvrun).In Scatter/ Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the applica- tion should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
8	bblerr	Babble Error (BblErr) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core..This bit can be set only by the core and the applica- tion should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
7	xacterr	Transaction Error (XactErr) Indicates one of the following errors occurred on the USB. CRC check failure Timeout Bit stuff error False EOP In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the applica- tion should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
6	nyet	NYET Response Received Interrupt (NYET) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value: Description** 0x0: INACTIVE 0x1: ACTIVE	RW	0x0
5	ack	ACK Response Received/Transmitted Interrupt (ACK) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value: Description** 0x0: INACTIVE 0x1: ACTIVE	RW	0x0
4	nak	NAK Response Received Interrupt (NAK) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value: Description** 0x0: INACTIVE 0x1: ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
3	stall	STALL Response Received Interrupt (STALL) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
2	ahberr	AHB Error (AHBErr) This is generated only in Internal DMA mode when there is an AHB error during AHB read/write. The application can read the corresponding channel's DMA address register to get the error address. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
1	chhltd	Channel Halted (ChHltd) In non Scatter/Gather DMA mode, it indicates the transfer completed abnormally either because of any USB transaction error or in response to disable request by the application or because of a completed transfer. in Scatter/gather DMA mode, this indicates that transfer completed due to any of the following . EOL being set in descriptor . AHB error . Excessive transaction errors . Babble . Stall **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
0	xfercompl	Transfer Completed (XferCompl) Transfer completed normally without any errors.This bit can be set only by the core and the application should write 1 to clear it. For Scatter/Gather DMA mode, it indicates that current descriptor processing got completed with IOC bit set in its descriptor. In non Scatter/Gather DMA mode, it indicates that Transfer completed normally without any errors. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

hcintmsk13

Host Channel 13 Interrupt Mask Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB006AC
i_usbotg_1_hostgrp	0xFFB40400	0xFFB406AC

Offset: 0x2AC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		frm_lst_rollintrmsk RW 0x0	Reserved	bnaintrmsk RW 0x0	datatglerrmsk RW 0x0	frmovrunmsk RW 0x0	bblerrmsk RW 0x0	xacterrmsk RW 0x0	nyetmsk RW 0x0	ackmsk RW 0x0	nakmsk RW 0x0	stallmsk RW 0x0	ahberrmsk RW 0x0	chhltdmsk RW 0x0	xfercomplmsk RW 0x0

hcintmsk13 Fields

Bit	Name	Description	Access	Reset
13	frm_lst_rollintrmsk	Framelist rollover interrupt Mask register(FRM_LST_ROLLIntrMsk) This bit is valid only when Scatter/Gather DMA mode is enabled. **Value** — **Description** 0x0 — MASK 0x1 — NOMASK	RW	0x0

Bit	Name	Description	Access	Reset
11	bnaintrmsk	BNA (Buffer Not Available) Interrupt mask register (BNAIntrMsk) This bit is valid only when Scatter/Gather DMA mode is enabled. Value: 0x0 MASK; 0x1 NOMASK	RW	0x0
10	datatglerrmsk	Data Toggle Error Mask (DataTglErrMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
9	frmovrunmsk	Frame Overrun Mask (FrmOvrunMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
8	bblerrmsk	Babble Error Mask (BblErrMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
7	xacterrmsk	Transaction Error Mask (XactErrMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0

Bit	Name	Description	Access	Reset
6	nyetmsk	NYET Response Received Interrupt Mask (NyetMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
5	ackmsk	ACK Response Received/Transmitted Interrupt Mask (AckMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
4	nakmsk	NAK Response Received Interrupt Mask (NakMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
3	stallmsk	STALL Response Received Interrupt Mask (StallMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0

Bit	Name	Description	Access	Reset
2	ahberrmsk	AHB Error Mask (AHBErrMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn. **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0
1	chhltdmsk	Channel Halted Mask (ChHltdMsk) **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0
0	xfercomplmsk	Transfer Completed Mask (XferComplMsk) **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0

hctsiz13

Host Channel 13 Transfer Size Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB006B0
i_usbotg_1_hostgrp	0xFFB40400	0xFFB406B0

Offset: 0x2B0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dopng RW 0x0	pid RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
xfersize RW 0x0															

hctsiz13 Fields

Bit	Name	Description	Access	Reset
31	dopng	Do Ping (DoPng) This bit is used only For OUT transfers. Setting this field to 1 directs the host to do PING protocol. Note: Do not Set this bit For IN transfers. If this bit is Set For for IN transfers it disables the channel. **Value** — **Description** 0x0 — NOPING 0x1 — PING	RW	0x0

Bit	Name	Description	Access	Reset
30:29	pid	PID (Pid) The application programs this field with the type of PID to use For the initial transaction. The host maintains this field For the rest of the transfer. 2'b00: DATA0 2'b01: DATA2 2'b10: DATA1 2'b11: MDATA (non-control)/SETUP (control) **Value** **Description** 0x0 DATA0 0x1 DATA2 0x2 DATA1 0x3 MDATA	RW	0x0
28:19	pktcnt	Packet Count (PktCnt) This field is programmed by the application with the expected number of packets to be transmitted (OUT) or received (IN). The host decrements this count on every successful transmission or reception of an OUT/IN packet. Once this count reaches zero, the application is interrupted to indicate normal completion. The width of this counter is specified as Width of Packet Counters	RW	0x0
18:0	xfersize	Transfer Size (XferSize) For an OUT, this field is the number of data bytes the host sends during the transfer. For an IN, this field is the buffer size that the application has Reserved For the transfer. The application is expected to program this field as an integer multiple of the maximum packet size For IN transactions (periodic and non-periodic). The width of this counter is specified as Width of Transfer Size Counters	RW	0x0

hcdma13

Host Channel 13 DMA Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB006B4
i_usbotg_1_hostgrp	0xFFB40400	0xFFB406B4

Offset: 0x2B4

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcdma13 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
hcdma13 RW 0x0															

hcdma13 Fields

Bit	Name	Description	Access	Reset
31:0	hcdma13	Buffer DMA Mode: [31:0] DMA Address (DMAAddr) This field holds the start address in the external memory from which the data for the endpoint must be fetched or to which it must be stored. This register is incremented on every AHB transac- tion. Scatter-Gather DMA (DescDMA) Mode: [31:9] (Non Isoc) Non-Isochronous: [31:N] (Isoc) Isochronous: This field holds the start address of the 512 bytes page. The first descriptor in the list should be located in this address. The first	RW	0x0

Bit	Name	Description	Access	Reset
		descriptor may be or may not be ready. The core starts processing the list from the CTD value. This field holds the address of the 2*(nTD+1) bytes of locations in which the isochronous descriptors are present where N is based on nTD as per Table below [31:N] Base Address [N-1:3] Offset [2:0] 000 HS ISOC nTD N 7 6 15 7 31 8 63 9 127 10 255 11 FS ISOC nTD N 1 4 3 5 7 6 15 7 31 8 63 9 [N-1:3] (Isoc): [8:3] (Non Isoc): Current Transfer Desc(CTD): Non Isochronous: This value is in terms of number of descriptors. The values can be from 0 to 63. 0 - 1 descriptor. 63 - 64 descriptors. This field indicates the current descriptor processed in the list. This field is updated both by application and the core. For example, if the application enables the channel after programming CTD=5, then the core will start processing the 6th descriptor. The address is obtained		

Bit	Name	Description	Access	Reset
		by adding a value of (8bytes*5=) 40(decimal) to DMAAddr. Isochronous: CTD for isochronous is based on the current frame/(micro)frame value. Need to be set to zero by application.Scatter-Gather DMA (DescDMA) Mode: [31:9] (Non Isoc) Non-Isochronous: [31:N] (Isoc) Isochronous: This field holds the start address of the 512 bytes page. The first descriptor in the list should be located in this address. The first descriptor may be or may not be ready. The core starts processing the list from the CTD value. This field holds the address of the 2*(nTD+1) bytes of locations in which the isochronous descriptors are present where N is based on nTD as per Table below [31:N] Base Address [N-1:3] Offset [2:0] 000 HS ISOC nTD N 7 6 15 7 31 8 63 9 127 10 255 11 FS ISOC nTD N 1 4 3 5 7 6 15 7 31 8 63 9 [N-1:3] (Isoc): [8:3] (Non Isoc): Current Transfer Desc(CTD):		

Bit	Name	Description	Access	Reset
		Non Isochronous: This value is in terms of number of descriptors. The values can be from 0 to 63. 0 - 1 descriptor. 63 - 64 descriptors. This field indicates the current descriptor processed in the list. This field is updated both by application and the core. For example, if the application enables the channel after programming CTD=5, then the core will start processing the 6th descriptor. The address is obtained by adding a value of (8bytes*5=) 40(decimal) to DMAAddr. Isochronous: CTD for isochronous is based on the current frame/(micro)frame value. Need to be set to zero by application.		

hcdmab13

Host Channel 13 DMA Buffer Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB006BC
i_usbotg_1_hostgrp	0xFFB40400	0xFFB406BC

Offset: 0x2BC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcdmab13 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
hcdmab13 RW 0x0															

hcdmab13 Fields

Bit	Name	Description	Access	Reset
31:0	hcdmab13	Holds the current buffer address. This register is updated as and when the data transfer for the corresponding end point is in progress. This register is present only in Scatter/Gather DMA mode. Otherwise this field is reserved.	RW	0x0

hcchar14

Host Channel 14 Characteristics Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB006C0
i_usbotg_1_hostgrp	0xFFB40400	0xFFB406C0

Offset: 0x2C0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
chena RW 0x0	chdis RW 0x0	oddfrm RW 0x0	devaddr RW 0x0							ec RW 0x0		eptype RW 0x0		lspddev RW 0x0	Reserved
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
epdir RW 0x0	epnum RW 0x0				mps RW 0x0										

hcchar14 Fields

Bit	Name	Description	Access	Reset
31	chena	Channel Enable (ChEna) When Scatter/Gather mode is enabled 1'b0: Indicates that the descriptor structure is not yet ready. 1'b1: Indicates that the descriptor structure and data buffer with data is setup and this channel can access the descriptor. When Scatter/Gather mode is disabled This field is set by the application and cleared by the OTG host. 1'b0: Channel disabled 1'b1: Channel enabled **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
30	chdis	Channel Disable (ChDis) The application sets this bit to stop transmitting/receiving data on a channel, even before the transfer For that channel is complete. The application must wait For the Channel Disabled interrupt before treating the channel as disabled. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
29	oddfrm	Odd Frame (OddFrm) This field is set (reset) by the application to indicate that the OTG host must perform a transfer in an odd (micro)frame. This field is applicable for only periodic (isochronous and interrupt) transactions. 1'b0: Even (micro)frame 1'b1: Odd (micro)frame	RW	0x0
28:22	devaddr	Device Address (DevAddr) This field selects the specific device serving as the data source or sink.	RW	0x0

Bit	Name	Description	Access	Reset
21:20	ec	Multi Count (MC) / Error Count (EC) When the Split Enable bit of the Host Channel-n Split Control register (HCSPLTn.SpltEna) is reset (1'b0), this field indicates to the host the number of transactions that must be executed per microframe For this periodic endpoint. For non periodic transfers, this field is used only in DMA mode, and specifies the number packets to be fetched For this channel before the internal DMA engine changes arbitration. 2'b00: Reserved This field yields undefined results. 2'b01: 1 transaction 2'b10: 2 transactions to be issued For this endpoint per microframe 2'b11: 3 transactions to be issued For this endpoint per microframe When HCSPLTn.SpltEna is Set (1'b1), this field indicates the number of immediate retries to be performed For a periodic split transactions on transaction errors. This field must be Set to at least 2'b01. **Value** **Description** 0x0 RESERVED 0x1 TRANSONE 0x2 TRANSTWO 0x3 TRANSTHREE	RW	0x0

Bit	Name	Description	Access	Reset
19:18	eptype	Endpoint Type (EPType) Indicates the transfer type selected. 2'b00: Control 2'b01: Isochronous 2'b10: Bulk 2'b11: Interrupt **Value** / **Description** 0x0 — CTRL 0x1 — ISOC 0x2 — BULK 0x3 — INTERR	RW	0x0
17	lspddev	Low-Speed Device (LSpdDev) This field is Set by the application to indicate that this channel is communicating to a low-speed device. **Value** / **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
15	epdir	Endpoint Direction (EPDir) Indicates whether the transaction is IN or OUT. 1'b0: OUT 1'b1: IN **Value** / **Description** 0x0 — OUT 0x1 — IN	RW	0x0

Bit	Name	Description	Access	Reset
14:11	epnum	Endpoint Number (EPNum) Indicates the endpoint number on the device serving as the data source or sink. Value — Description 0xa — ENDPT10 0xb — ENDPT11 0xc — ENDPT12 0xd — ENDPT13 0xe — ENDPT14 0xf — ENDPT15 0x0 — ENDPT0 0x1 — ENDPT1 0x2 — ENDPT2 0x3 — ENDPT3 0x4 — ENDPT4 0x5 — ENDPT5 0x6 — ENDPT6 0x7 — ENDPT7 0x8 — ENDPT8 0x9 — ENDPT9	RW	0x0
10:0	mps	Maximum Packet Size (MPS) Indicates the maximum packet size of the associated endpoint.	RW	0x0

hcsplt14

Host Channel 14 Split Control Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB006C4
i_usbotg_1_hostgrp	0xFFB40400	0xFFB406C4

Offset: 0x2C4

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
spltena RW 0x0	Reserved														compsplt RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
xactpos RW 0x0	hubaddr RW 0x0								prtaddr RW 0x0						

hcsplt14 Fields

Bit	Name	Description	Access	Reset
31	spltena	Split Enable (SpltEna) The application sets this field to indicate that this channel is enabled to perform split transactions. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
16	compsplt	Do Complete Split (CompSplt) The application sets this field to request the OTG host to perform a complete split transaction. **Value** — **Description** 0x0 — NOSPLIT 0x1 — SPLIT	RW	0x0

Bit	Name	Description	Access	Reset
15:14	xactpos	Transaction Position (XactPos) This field is used to determine whether to send all, first, middle, or last payloads with each OUT transaction. 2'b11: All. This is the entire data payload is of this transaction (which is less than or equal to 188 bytes). 2'b10: Begin. This is the first data payload of this transaction (which is larger than 188 bytes). 2'b00: Mid. This is the middle payload of this transaction (which is larger than 188 bytes). 2'b01: End. This is the last payload of this transaction (which is larger than 188 bytes). **Value** **Description** 0x0 MIDDLE 0x1 END 0x2 BEGIN 0x3 ALL	RW	0x0
13:7	hubaddr	Hub Address (HubAddr) This field holds the device address of the transaction translator's hub.	RW	0x0
6:0	prtaddr	Port Address (PrtAddr) This field is the port number of the recipient transaction translator.	RW	0x0

hcint14

Host Channel 14 Interrupt Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB006C8

Module Instance	Base Address	Register Address
i_usbotg_1_hostgrp	0xFFB40400	0xFFB406C8

Offset: 0x2C8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		desc_lst_rollintr RW 0x0	xcs_xact_err RW 0x0	bnaintr RW 0x0	datatglerr RW 0x0	frmovrun RW 0x0	bblerr RW 0x0	xacterr RW 0x0	nyet RW 0x0	ack RW 0x0	nak RW 0x0	stall RW 0x0	ahberr RW 0x0	chhltd RW 0x0	xfercompl RW 0x0

hcint14 Fields

Bit	Name	Description	Access	Reset
13	desc_lst_rollintr	Descriptor rollover interrupt (DESC_LST_ROLLIntr) This bit is valid only when Scatter/Gather DMA mode is enabled. The core sets this bit when the corresponding channel's descriptor list rolls over. For non Scatter/Gather DMA mode, this bit is reserved. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
12	xcs_xact_err	Excessive Transaction Error (XCS_XACT_ERR) This bit is valid only when Scatter/Gather DMA mode is enabled. The core sets this bit when 3 consecutive transaction errors occurred on the USB bus. XCS_XACT_ERR will not be generated for Isochronous channels. For non Scatter/Gather DMA mode, this bit is reserved. **Value** **Description** 0x0 INACTIVE 0x1 ACVTIVE	RW	0x0
11	bnaintr	BNA (Buffer Not Available) Interrupt (BNAIntr) This bit is valid only when Scatter/Gather DMA mode is enabled. The core generates this interrupt when the descriptor accessed is not ready for the Core to process. BNA will not be generated for Isochronous channels. For non Scatter/Gather DMA mode, this bit is reserved. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
10	datatglerr	Data Toggle Error (DataTglErr).This bit can be set only by the core and the application should write 1 to clear it.In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
9	frmovrun	Frame Overrun (FrmOvrun).In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
8	bblerr	Babble Error (BblErr) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core..This bit can be set only by the core and the application should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
7	xacterr	Transaction Error (XactErr) Indicates one of the following errors occurred on the USB. CRC check failure Timeout Bit stuff error False EOP In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
6	nyet	NYET Response Received Interrupt (NYET) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
5	ack	ACK Response Received/Transmitted Interrupt (ACK) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
4	nak	NAK Response Received Interrupt (NAK) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
3	stall	STALL Response Received Interrupt (STALL) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
2	ahberr	AHB Error (AHBErr) This is generated only in Internal DMA mode when there is an AHB error during AHB read/write. The application can read the corresponding channel's DMA address register to get the error address. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
1	chhltd	Channel Halted (ChHltd) In non Scatter/Gather DMA mode, it indicates the transfer completed abnormally either because of any USB transaction error or in response to disable request by the application or because of a completed transfer. in Scatter/gather DMA mode, this indicates that transfer completed due to any of the following . EOL being set in descriptor . AHB error . Excessive transaction errors . Babble . Stall **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
0	xfercompl	Transfer Completed (XferCompl) Transfer completed normally without any errors.This bit can be set only by the core and the application should write 1 to clear it. For Scatter/Gather DMA mode, it indicates that current descriptor processing got completed with IOC bit set in its descriptor. In non Scatter/Gather DMA mode, it indicates that Transfer completed normally without any errors. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

hcintmsk14

Host Channel 14 Interrupt Mask Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB006CC
i_usbotg_1_hostgrp	0xFFB40400	0xFFB406CC

Offset: 0x2CC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		frm_lst_rollintrmsk RW 0x0	Reserved	bnaintrmsk RW 0x0	datatglerrmsk RW 0x0	frmovrunmsk RW 0x0	bblerrmsk RW 0x0	xacterrmsk RW 0x0	nyetmsk RW 0x0	ackmsk RW 0x0	nakmsk RW 0x0	stallmsk RW 0x0	ahberrmsk RW 0x0	chhltdmsk RW 0x0	xfercomplmsk RW 0x0

hcintmsk14 Fields

Bit	Name	Description	Access	Reset
13	frm_lst_rollintrmsk	Framelist rollover interrupt Mask register(FRM_LST_ROLLIntrMsk) This bit is valid only when Scatter/Gather DMA mode is enabled. **Value** — **Description** 0x0 — MASK 0x1 — NOMASK	RW	0x0

Bit	Name	Description	Access	Reset
11	bnaintrmsk	BNA (Buffer Not Available) Interrupt mask register (BNAIntrMsk) This bit is valid only when Scatter/Gather DMA mode is enabled. **Value** — **Description** 0x0 — MASK 0x1 — NOMASK	RW	0x0
10	datatglerrmsk	Data Toggle Error Mask (DataTglErrMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
9	frmovrunmsk	Frame Overrun Mask (FrmOvrunMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
8	bblerrmsk	Babble Error Mask (BblErrMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
7	xacterrmsk	Transaction Error Mask (XactErrMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0

Bit	Name	Description	Access	Reset
6	nyetmsk	NYET Response Received Interrupt Mask (NyetMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
5	ackmsk	ACK Response Received/Transmitted Interrupt Mask (AckMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
4	nakmsk	NAK Response Received Interrupt Mask (NakMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
3	stallmsk	STALL Response Received Interrupt Mask (StallMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0

Bit	Name	Description	Access	Reset
2	ahberrmsk	AHB Error Mask (AHBErrMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn. **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0
1	chhltdmsk	Channel Halted Mask (ChHltdMsk) **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0
0	xfercomplmsk	Transfer Completed Mask (XferComplMsk) **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0

hctsiz14

Host Channel 14 Transfer Size Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB006D0
i_usbotg_1_hostgrp	0xFFB40400	0xFFB406D0

Offset: 0x2D0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dopng RW 0x0	pid RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
xfersize RW 0x0															

hctsiz14 Fields

Bit	Name	Description	Access	Reset
31	dopng	Do Ping (DoPng) This bit is used only For OUT transfers. Setting this field to 1 directs the host to do PING protocol. Note: Do not Set this bit For IN transfers. If this bit is Set For for IN transfers it disables the channel. **Value** — **Description** 0x0 — NOPING 0x1 — PING	RW	0x0

Bit	Name	Description	Access	Reset
30:29	pid	PID (Pid) The application programs this field with the type of PID to use For the initial transaction. The host maintains this field For the rest of the transfer. 2'b00: DATA0 2'b01: DATA2 2'b10: DATA1 2'b11: MDATA (non-control)/SETUP (control) **Value** **Description** 0x0 DATA0 0x1 DATA2 0x2 DATA1 0x3 MDATA	RW	0x0
28:19	pktcnt	Packet Count (PktCnt) This field is programmed by the application with the expected number of packets to be transmitted (OUT) or received (IN). The host decrements this count on every successful transmission or reception of an OUT/IN packet. Once this count reaches zero, the application is interrupted to indicate normal completion. The width of this counter is specified as Width of Packet Counters	RW	0x0
18:0	xfersize	Transfer Size (XferSize) For an OUT, this field is the number of data bytes the host sends during the transfer. For an IN, this field is the buffer size that the application has Reserved For the transfer. The application is expected to program this field as an integer multiple of the maximum packet size For IN transactions (periodic and non-periodic). The width of this counter is specified as Width of Transfer Size Counters	RW	0x0

hcdma14

Host Channel 14 DMA Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB006D4
i_usbotg_1_hostgrp	0xFFB40400	0xFFB406D4

Offset: 0x2D4

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcdma14 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
hcdma14 RW 0x0															

hcdma14 Fields

Bit	Name	Description	Access	Reset
31:0	hcdma14	Buffer DMA Mode: [31:0] DMA Address (DMAAddr) This field holds the start address in the external memory from which the data for the endpoint must be fetched or to which it must be stored. This register is incremented on every AHB transac- tion. Scatter-Gather DMA (DescDMA) Mode: [31:9] (Non Isoc) Non-Isochronous: [31:N] (Isoc) Isochronous: This field holds the start address of the 512 bytes page. The first descriptor in the list should be located in this address. The first	RW	0x0

Bit	Name	Description	Access	Reset
		descriptor may be or may not be ready. The core starts processing the list from the CTD value. This field holds the address of the 2*(nTD+1) bytes of locations in which the isochronous descriptors are present where N is based on nTD as per Table below [31:N] Base Address [N-1:3] Offset [2:0] 000 HS ISOC nTD N 7 6 15 7 31 8 63 9 127 10 255 11 FS ISOC nTD N 1 4 3 5 7 6 15 7 31 8 63 9 [N-1:3] (Isoc): [8:3] (Non Isoc): Current Transfer Desc(CTD): Non Isochronous: This value is in terms of number of descriptors. The values can be from 0 to 63. 0 - 1 descriptor. 63 - 64 descriptors. This field indicates the current descriptor processed in the list. This field is updated both by application and the core. For example, if the application enables the channel after programming CTD=5, then the core will start processing the 6th descriptor. The address is obtained		

Bit	Name	Description	Access	Reset
		by adding a value of (8bytes*5=) 40(decimal) to DMAAddr. Isochronous: CTD for isochronous is based on the current frame/(micro)frame value. Need to be set to zero by application.Scatter-Gather DMA (DescDMA) Mode: [31:9] (Non Isoc) Non-Isochronous: [31:N] (Isoc) Isochronous: This field holds the start address of the 512 bytes page. The first descriptor in the list should be located in this address. The first descriptor may be or may not be ready. The core starts processing the list from the CTD value. This field holds the address of the 2*(nTD+1) bytes of locations in which the isochronous descriptors are present where N is based on nTD as per Table below [31:N] Base Address [N-1:3] Offset [2:0] 000 HS ISOC nTD N 7 6 15 7 31 8 63 9 127 10 255 11 FS ISOC nTD N 1 4 3 5 7 6 15 7 31 8 63 9 [N-1:3] (Isoc): [8:3] (Non Isoc): Current Transfer Desc(CTD):		

Bit	Name	Description	Access	Reset
		Non Isochronous: This value is in terms of number of descriptors. The values can be from 0 to 63. 0 - 1 descriptor. 63 - 64 descriptors. This field indicates the current descriptor processed in the list. This field is updated both by application and the core. For example, if the application enables the channel after programming CTD=5, then the core will start processing the 6th descriptor. The address is obtained by adding a value of (8bytes*5=) 40(decimal) to DMAAddr. Isochronous: CTD for isochronous is based on the current frame/(micro)frame value. Need to be set to zero by application.		

hcdmab14

Host Channel 14 DMA Buffer Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB006DC
i_usbotg_1_hostgrp	0xFFB40400	0xFFB406DC

Offset: 0x2DC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcdmab14 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
hcdmab14 RW 0x0															

hcdmab14 Fields

Bit	Name	Description	Access	Reset
31:0	hcdmab14	Holds the current buffer address. This register is updated as and when the data transfer for the corresponding end point is in progress. This register is present only in Scatter/Gather DMA mode. Otherwise this field is reserved.	RW	0x0

hcchar15

Host Channel 15 Characteristics Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB006E0
i_usbotg_1_hostgrp	0xFFB40400	0xFFB406E0

Offset: 0x2E0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
chena RW 0x0	chdis RW 0x0	oddfrm RW 0x0	devaddr RW 0x0							ec RW 0x0		eptype RW 0x0		lspddev RW 0x0	Reserved
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
epdir RW 0x0	epnum RW 0x0				mps RW 0x0										

hcchar15 Fields

Bit	Name	Description	Access	Reset
31	chena	Channel Enable (ChEna) When Scatter/Gather mode is enabled 1'b0: Indicates that the descriptor structure is not yet ready. 1'b1: Indicates that the descriptor structure and data buffer with data is setup and this channel can access the descriptor. When Scatter/Gather mode is disabled This field is set by the applica-tion and cleared by the OTG host. 1'b0: Channel disabled 1'b1: Channel enabled **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
30	chdis	Channel Disable (ChDis) The application sets this bit to stop transmitting/receiving data on a channel, even before the transfer For that channel is complete. The application must wait For the Channel Disabled interrupt before treating the channel as disabled. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
29	oddfrm	Odd Frame (OddFrm) This field is set (reset) by the application to indicate that the OTG host must perform a transfer in an odd (micro)frame. This field is applicable for only periodic (isochronous and interrupt) transactions. 1'b0: Even (micro)frame 1'b1: Odd (micro)frame	RW	0x0
28:22	devaddr	Device Address (DevAddr) This field selects the specific device serving as the data source or sink.	RW	0x0

Bit	Name	Description	Access	Reset
21:20	ec	Multi Count (MC) / Error Count (EC) When the Split Enable bit of the Host Channel-n Split Control register (HCSPLTn.SpltEna) is reset (1'b0), this field indicates to the host the number of transactions that must be executed per microframe For this periodic endpoint. For non periodic transfers, this field is used only in DMA mode, and specifies the number packets to be fetched For this channel before the internal DMA engine changes arbitration. 2'b00: Reserved This field yields undefined results. 2'b01: 1 transaction 2'b10: 2 transactions to be issued For this endpoint per microframe 2'b11: 3 transactions to be issued For this endpoint per microframe When HCSPLTn.SpltEna is Set (1'b1), this field indicates the number of immediate retries to be performed For a periodic split transactions on transaction errors. This field must be Set to at least 2'b01. **Value** **Description** 0x0 RESERVED 0x1 TRANSONE 0x2 TRANSTWO 0x3 TRANSTHREE	RW	0x0

Bit	Name	Description	Access	Reset
19:18	eptype	Endpoint Type (EPType) Indicates the transfer type selected. 2'b00: Control 2'b01: Isochronous 2'b10: Bulk 2'b11: Interrupt **Value** — **Description** 0x0 — CTRL 0x1 — ISOC 0x2 — BULK 0x3 — INTERR	RW	0x0
17	lspddev	Low-Speed Device (LSpdDev) This field is Set by the application to indicate that this channel is communicating to a low-speed device. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
15	epdir	Endpoint Direction (EPDir) Indicates whether the transaction is IN or OUT. 1'b0: OUT 1'b1: IN **Value** — **Description** 0x0 — OUT 0x1 — IN	RW	0x0

Bit	Name	Description	Access	Reset
14:11	epnum	Endpoint Number (EPNum) Indicates the endpoint number on the device serving as the data source or sink. Value — Description 0xa — ENDPT10 0xb — ENDPT11 0xc — ENDPT12 0xd — ENDPT13 0xe — ENDPT14 0xf — ENDPT15 0x0 — ENDPT0 0x1 — ENDPT1 0x2 — ENDPT2 0x3 — ENDPT3 0x4 — ENDPT4 0x5 — ENDPT5 0x6 — ENDPT6 0x7 — ENDPT7 0x8 — ENDPT8 0x9 — ENDPT9	RW	0x0
10:0	mps	Maximum Packet Size (MPS) Indicates the maximum packet size of the associated endpoint.	RW	0x0

hcsplt15

Host Channel 15 Split Control Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB006E4
i_usbotg_1_hostgrp	0xFFB40400	0xFFB406E4

Offset: 0x2E4

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
spltena RW 0x0	Reserved														compsplt RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
xactpos RW 0x0	hubaddr RW 0x0								prtaddr RW 0x0						

hcsplt15 Fields

Bit	Name	Description	Access	Reset
31	spltena	Split Enable (SpltEna) The application sets this field to indicate that this channel is enabled to perform split transactions. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
16	compsplt	Do Complete Split (CompSplt) The application sets this field to request the OTG host to perform a complete split transaction. **Value** — **Description** 0x0 — NOSPLIT 0x1 — SPLIT	RW	0x0

Bit	Name	Description	Access	Reset
15:14	xactpos	Transaction Position (XactPos) This field is used to determine whether to send all, first, middle, or last payloads with each OUT transaction. 2'b11: All. This is the entire data payload is of this transaction (which is less than or equal to 188 bytes). 2'b10: Begin. This is the first data payload of this transaction (which is larger than 188 bytes). 2'b00: Mid. This is the middle payload of this transaction (which is larger than 188 bytes). 2'b01: End. This is the last payload of this transaction (which is larger than 188 bytes). **Value** **Description** 0x0 MIDDLE 0x1 END 0x2 BEGIN 0x3 ALL	RW	0x0
13:7	hubaddr	Hub Address (HubAddr) This field holds the device address of the transaction translator's hub.	RW	0x0
6:0	prtaddr	Port Address (PrtAddr) This field is the port number of the recipient transaction translator.	RW	0x0

hcint15

Host Channel 15 Interrupt Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB006E8

Module Instance	Base Address	Register Address
i_usbotg_1_hostgrp	0xFFB40400	0xFFB406E8

Offset: 0x2E8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		desc_lst_rollintr RW 0x0	xcs_xact_err RW 0x0	bnaintr RW 0x0	datatglerr RW 0x0	frmovrun RW 0x0	bblerr RW 0x0	xacterr RW 0x0	nyet RW 0x0	ack RW 0x0	nak RW 0x0	stall RW 0x0	ahberr RW 0x0	chhltd RW 0x0	xfercompl RW 0x0

hcint15 Fields

Bit	Name	Description	Access	Reset
13	desc_lst_rollintr	Descriptor rollover interrupt (DESC_LST_ROLLIntr) This bit is valid only when Scatter/Gather DMA mode is enabled. The core sets this bit when the corresponding channel's descriptor list rolls over. For non Scatter/Gather DMA mode, this bit is reserved. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
12	xcs_xact_err	Excessive Transaction Error (XCS_XACT_ERR) This bit is valid only when Scatter/Gather DMA mode is enabled. The core sets this bit when 3 consecutive transaction errors occurred on the USB bus. XCS_XACT_ERR will not be generated for Isochronous channels. For non Scatter/Gather DMA mode, this bit is reserved. **Value** **Description** 0x0 INACTIVE 0x1 ACVTIVE	RW	0x0
11	bnaintr	BNA (Buffer Not Available) Interrupt (BNAIntr) This bit is valid only when Scatter/Gather DMA mode is enabled. The core generates this interrupt when the descriptor accessed is not ready for the Core to process. BNA will not be generated for Isochronous channels. For non Scatter/Gather DMA mode, this bit is reserved. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
10	datatglerr	Data Toggle Error (DataTglErr).This bit can be set only by the core and the application should write 1 to clear it.In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
9	frmovrun	Frame Overrun (FrmOvrun).In Scatter/ Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the applica- tion should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
8	bblerr	Babble Error (BblErr) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core..This bit can be set only by the core and the applica- tion should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
7	xacterr	Transaction Error (XactErr) Indicates one of the following errors occurred on the USB. CRC check failure Timeout Bit stuff error False EOP In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the applica- tion should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
6	nyet	NYET Response Received Interrupt (NYET) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
5	ack	ACK Response Received/Transmitted Interrupt (ACK) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
4	nak	NAK Response Received Interrupt (NAK) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
3	stall	STALL Response Received Interrupt (STALL) In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.This bit can be set only by the core and the application should write 1 to clear it. **Value** / **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
2	ahberr	AHB Error (AHBErr) This is generated only in Internal DMA mode when there is an AHB error during AHB read/write. The application can read the corresponding channel's DMA address register to get the error address. **Value** / **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
1	chhltd	Channel Halted (ChHltd) In non Scatter/Gather DMA mode, it indicates the transfer completed abnormally either because of any USB transaction error or in response to disable request by the application or because of a completed transfer. in Scatter/gather DMA mode, this indicates that transfer completed due to any of the following . EOL being set in descriptor . AHB error . Excessive transaction errors . Babble . Stall **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
0	xfercompl	Transfer Completed (XferCompl) Transfer completed normally without any errors.This bit can be set only by the core and the application should write 1 to clear it. For Scatter/Gather DMA mode, it indicates that current descriptor processing got completed with IOC bit set in its descriptor. In non Scatter/Gather DMA mode, it indicates that Transfer completed normally without any errors. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

hcintmsk15

Host Channel 15 Interrupt Mask Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB006EC
i_usbotg_1_hostgrp	0xFFB40400	0xFFB406EC

Offset: 0x2EC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		frm_lst_rollintrmsk RW 0x0	Reserved	bnaintrmsk RW 0x0	datatglerrmsk RW 0x0	frmovrunmsk RW 0x0	bblerrmsk RW 0x0	xacterrmsk RW 0x0	nyetmsk RW 0x0	ackmsk RW 0x0	nakmsk RW 0x0	stallmsk RW 0x0	ahberrmsk RW 0x0	chhltdmsk RW 0x0	xfercomplmsk RW 0x0

hcintmsk15 Fields

Bit	Name	Description	Access	Reset
13	frm_lst_rollintrmsk	Framelist rollover interrupt Mask register(FRM_LST_ROLLIntrMsk) This bit is valid only when Scatter/Gather DMA mode is enabled. **Value** — **Description** 0x0 — MASK 0x1 — NOMASK	RW	0x0

Bit	Name	Description	Access	Reset
11	bnaintrmsk	BNA (Buffer Not Available) Interrupt mask register (BNAIntrMsk) This bit is valid only when Scatter/Gather DMA mode is enabled. **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0
10	datatglerrmsk	Data Toggle Error Mask (DataTglErrMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
9	frmovrunmsk	Frame Overrun Mask (FrmOvrunMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
8	bblerrmsk	Babble Error Mask (BblErrMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
7	xacterrmsk	Transaction Error Mask (XactErrMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0

Bit	Name	Description	Access	Reset
6	nyetmsk	NYET Response Received Interrupt Mask (NyetMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
5	ackmsk	ACK Response Received/Transmitted Interrupt Mask (AckMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
4	nakmsk	NAK Response Received Interrupt Mask (NakMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0
3	stallmsk	STALL Response Received Interrupt Mask (StallMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn.	RW	0x0

Bit	Name	Description	Access	Reset
2	ahberrmsk	AHB Error Mask (AHBErrMsk) In scatter/gather DMA mode for host, interrupts will not be generated due to the corresponding bits set in HCINTn. **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0
1	chhltdmsk	Channel Halted Mask (ChHltdMsk) **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0
0	xfercomplmsk	Transfer Completed Mask (XferComplMsk) **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0

hctsiz15

Host Channel 15 Transfer Size Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB006F0
i_usbotg_1_hostgrp	0xFFB40400	0xFFB406F0

Offset: 0x2F0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dopng RW 0x0	pid RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
xfersize RW 0x0															

hctsiz15 Fields

Bit	Name	Description	Access	Reset
31	dopng	Do Ping (DoPng) This bit is used only For OUT transfers. Setting this field to 1 directs the host to do PING protocol. Note: Do not Set this bit For IN transfers. If this bit is Set For for IN transfers it disables the channel. **Value** — **Description** 0x0 — NOPING 0x1 — PING	RW	0x0

Bit	Name	Description	Access	Reset
30:29	pid	PID (Pid) The application programs this field with the type of PID to use For the initial transaction. The host maintains this field For the rest of the transfer. 2'b00: DATA0 2'b01: DATA2 2'b10: DATA1 2'b11: MDATA (non-control)/SETUP (control) **Value** **Description** 0x0 DATA0 0x1 DATA2 0x2 DATA1 0x3 MDATA	RW	0x0
28:19	pktcnt	Packet Count (PktCnt) This field is programmed by the application with the expected number of packets to be transmitted (OUT) or received (IN). The host decrements this count on every successful transmission or reception of an OUT/IN packet. Once this count reaches zero, the application is interrupted to indicate normal completion. The width of this counter is specified as Width of Packet Counters	RW	0x0
18:0	xfersize	Transfer Size (XferSize) For an OUT, this field is the number of data bytes the host sends during the transfer. For an IN, this field is the buffer size that the application has Reserved For the transfer. The application is expected to program this field as an integer multiple of the maximum packet size For IN transactions (periodic and non-periodic). The width of this counter is specified as Width of Transfer Size Counters	RW	0x0

hcdma15

Host Channel 15 DMA Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB006F4
i_usbotg_1_hostgrp	0xFFB40400	0xFFB406F4

Offset: 0x2F4

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcdma15 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
hcdma15 RW 0x0															

hcdma15 Fields

Bit	Name	Description	Access	Reset
31:0	hcdma15	Buffer DMA Mode: [31:0] DMA Address (DMAAddr) This field holds the start address in the external memory from which the data for the endpoint must be fetched or to which it must be stored. This register is incremented on every AHB transaction. Scatter-Gather DMA (DescDMA) Mode: [31:9] (Non Isoc) Non-Isochronous: [31:N] (Isoc) Isochronous: This field holds the start address of the 512 bytes page. The first descriptor in the list should be located in this address. The first descriptor may be or may not be ready. The core starts processing the list from the CTD value. This field holds the address of the 2*(nTD+1) bytes of locations in which the isochronous descriptors are present where N is based on nTD as per Table below	RW	0x0

Bit	Name	Description	Access	Reset
		[31:N] Base Address [N-1:3] Offset [2:0] 000 HS ISOC nTD N 7 6 15 7 31 8 63 9 127 10 255 11 FS ISOC nTD N 1 4 3 5 7 6 15 7 31 8 63 9 [N-1:3] (Isoc): [8:3] (Non Isoc): Current Transfer Desc(CTD): Non Isochronous: This value is in terms of number of descriptors. The values can be from 0 to 63. • 0 - 1 descriptor. • 63 - 64 descriptors. This field indicates the current descriptor processed in the list. This field is updated both by application and the core. For example, if the application enables the channel after programming CTD=5, then the core will start processing the 6th descriptor. The address is obtained by adding a value of (8bytes*5=) 40(decimal) to DMAAddr.		

Bit	Name	Description	Access	Reset
		Isochronous: CTD for isochronous is based on the current frame/ (micro)frame value. Need to be set to zero by application.Scatter-Gather DMA (DescDMA) Mode: [31:9] (Non Isoc) Non-Isochronous: [31:N] (Isoc) Isochronous: This field holds the start address of the 512 bytes page. The first descriptor in the list should be located in this address. The first descriptor may be or may not be ready. The core starts processing the list from the CTD value. This field holds the address of the 2*(nTD+1) bytes of locations in which the isochronous descriptors are present where N is based on nTD as per Table below [31:N] Base Address [N-1:3] Offset [2:0] 000 HS ISOC nTD N 7 6 15 7 31 8 63 9 127 10 255 11 FS ISOC nTD N 1 4 3 5 7 6 15 7 31 8 63 9		

Bit	Name	Description	Access	Reset
		[N-1:3] (Isoc): [8:3] (Non Isoc): Current Transfer Desc(CTD): Non Isochronous: This value is in terms of number of descriptors. The values can be from 0 to 63. • 0 - 1 descriptor. • 63 - 64 descriptors. This field indicates the current descriptor processed in the list. This field is updated both by application and the core. For example, if the application enables the channel after programming CTD=5, then the core will start processing the 6th descriptor. The address is obtained by adding a value of (8bytes*5=) 40(decimal) to DMAAddr. Isochronous: CTD for isochronous is based on the current frame/ (micro)frame value. Need to be set to zero by application.		

hcdmab15

Host Channel 15 DMA Buffer Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_hostgrp	0xFFB00400	0xFFB006FC
i_usbotg_1_hostgrp	0xFFB40400	0xFFB406FC

Offset: 0x2FC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
hcdmab15 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
hcdmab15 RW 0x0															

hcdmab15 Fields

Bit	Name	Description	Access	Reset
31:0	hcdmab15	Holds the current buffer address. This register is updated as and when the data transfer for the corresponding end point is in progress. This register is present only in Scatter/Gather DMA mode. Otherwise this field is reserved.	RW	0x0

usb_devgrp Address Map

Module Instance	Base Address	End Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00DFF
i_usbotg_1_devgrp	0xFFB40800	0xFFB40DFF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
dcfg on page 18-700	0x0	32	RW	0x8200000	Device Configuration Register
dctl on page 18-706	0x4	32	RW	0x2	Device Control Register
dsts on page 18-715	0x8	32	RO	0x2	Device Status Register

Register	Offset	Width	Access	Reset Value	Description
diepmsk on page 18-718	0x10	32	RW	0x0	Device IN Endpoint Common Interrupt Mask Register
doepmsk on page 18-721	0x14	32	RW	0x0	Device OUT Endpoint Common Interrupt Mask Register
daint on page 18-724	0x18	32	RO	0x0	Device All Endpoints Interrupt Register
daintmsk on page 18-730	0x1C	32	RW	0x0	Device All Endpoints Interrupt Mask Register
dvbusdis on page 18-736	0x28	32	RW	0x17D7	Device VBUS Discharge Time Register
dvbuspulse on page 18-737	0x2C	32	RW	0x5B8	Device VBUS Pulsing Time Register
dthrctl on page 18-738	0x30	32	RW	0xC100020	Device Threshold Control Register
diepempmsk on page 18-743	0x34	32	RW	0x0	Device IN Endpoint FIFO Empty Interrupt Mask Register
diepctl0 on page 18-746	0x100	32	RW	0x8000	Device Control IN Endpoint 0 Control Register
diepint0 on page 18-751	0x108	32	RW	0x80	Device IN Endpoint 0 Interrupt Register
dieptsiz0 on page 18-758	0x110	32	RW	0x0	Device IN Endpoint 0 Transfer Size Register
diepdma0 on page 18-760	0x114	32	RW	0x0	Device IN Endpoint 0 DMA Address Register
dtxfsts0 on page 18-761	0x118	32	RO	0x2000	Device IN Endpoint Transmit FIFO Status Register 0
diepdmab0 on page 18-762	0x11C	32	RO	0x0	Device IN Endpoint 16 Buffer Address Register

Register	Offset	Width	Access	Reset Value	Description
diepctl1 on page 18-763	0x120	32	RW	0x0	Device Control IN Endpoint 1 Control Register
diepint1 on page 18-772	0x128	32	RW	0x80	Device IN Endpoint 1 Interrupt Register
dieptsiz1 on page 18-779	0x130	32	RW	0x0	Device IN Endpoint 1 Transfer Size Register
diepdma1 on page 18-781	0x134	32	RW	0x0	Device IN Endpoint 1 DMA Address Register
dtxfsts1 on page 18-782	0x138	32	RO	0x2000	Device IN Endpoint Transmit FIFO Status Register 1
diepdmab1 on page 18-783	0x13C	32	RO	0x0	Device IN Endpoint 1 Buffer Address Register
diepctl2 on page 18-784	0x140	32	RW	0x0	Device Control IN Endpoint 2 Control Register
diepint2 on page 18-793	0x148	32	RW	0x80	Device IN Endpoint 2 Interrupt Register
dieptsiz2 on page 18-800	0x150	32	RW	0x0	Device IN Endpoint 2 Transfer Size Register
diepdma2 on page 18-802	0x154	32	RW	0x0	Device IN Endpoint 2 DMA Address Register
dtxfsts2 on page 18-803	0x158	32	RO	0x2000	Device IN Endpoint Transmit FIFO Status Register 2
diepdmab2 on page 18-804	0x15C	32	RO	0x0	Device IN Endpoint 2 Buffer Address Register
diepctl3 on page 18-805	0x160	32	RW	0x0	Device Control IN Endpoint 3 Control Register
diepint3 on page 18-814	0x168	32	RW	0x80	Device IN Endpoint 3 Interrupt Register
dieptsiz3 on page 18-821	0x170	32	RW	0x0	Device IN Endpoint 3 Transfer Size Register

Register	Offset	Width	Access	Reset Value	Description
diepdma3 on page 18-823	0x174	32	RW	0x0	Device IN Endpoint 3 DMA Address Register
dtxfsts3 on page 18-824	0x178	32	RO	0x2000	Device IN Endpoint Transmit FIFO Status Register 3
diepdmab3 on page 18-825	0x17C	32	RO	0x0	Device IN Endpoint 3 Buffer Address Register
diepctl4 on page 18-826	0x180	32	RW	0x0	Device Control IN Endpoint 4 Control Register
diepint4 on page 18-835	0x188	32	RW	0x80	Device IN Endpoint 4 Interrupt Register
dieptsiz4 on page 18-842	0x190	32	RW	0x0	Device IN Endpoint 4 Transfer Size Register
diepdma4 on page 18-844	0x194	32	RW	0x0	Device IN Endpoint 4 DMA Address Register
dtxfsts4 on page 18-845	0x198	32	RO	0x2000	Device IN Endpoint Transmit FIFO Status Register 4
diepdmab4 on page 18-846	0x19C	32	RO	0x0	Device IN Endpoint 4 Buffer Address Register
diepctl5 on page 18-847	0x1A0	32	RW	0x0	Device Control IN Endpoint 5 Control Register
diepint5 on page 18-856	0x1A8	32	RW	0x80	Device IN Endpoint 5 Interrupt Register
dieptsiz5 on page 18-863	0x1B0	32	RW	0x0	Device IN Endpoint 5 Transfer Size Register
diepdma5 on page 18-865	0x1B4	32	RW	0x0	Device IN Endpoint 5 DMA Address Register
dtxfsts5 on page 18-866	0x1B8	32	RO	0x2000	Device IN Endpoint Transmit FIFO Status Register 5
diepdmab5 on page 18-867	0x1BC	32	RO	0x0	Device IN Endpoint 5 Buffer Address Register

Register	Offset	Width	Access	Reset Value	Description
diepctl6 on page 18-868	0x1C0	32	RW	0x0	Device Control IN Endpoint 6 Control Register
diepint6 on page 18-877	0x1C8	32	RW	0x80	Device IN Endpoint 6 Interrupt Register
dieptsiz6 on page 18-884	0x1D0	32	RW	0x0	Device IN Endpoint 6 Transfer Size Register
diepdma6 on page 18-886	0x1D4	32	RW	0x0	Device IN Endpoint 6 DMA Address Register
dtxfsts6 on page 18-887	0x1D8	32	RO	0x2000	Device IN Endpoint Transmit FIFO Status Register 6
diepdmab6 on page 18-888	0x1DC	32	RO	0x0	Device IN Endpoint 6 Buffer Address Register
diepctl7 on page 18-889	0x1E0	32	RW	0x0	Device Control IN Endpoint 7 Control Register
diepint7 on page 18-898	0x1E8	32	RW	0x80	Device IN Endpoint 7 Interrupt Register
dieptsiz7 on page 18-905	0x1F0	32	RW	0x0	Device IN Endpoint 7 Transfer Size Register
diepdma7 on page 18-907	0x1F4	32	RW	0x0	Device IN Endpoint 7 DMA Address Register
dtxfsts7 on page 18-908	0x1F8	32	RO	0x2000	Device IN Endpoint Transmit FIFO Status Register 7
diepdmab7 on page 18-909	0x1FC	32	RO	0x0	Device IN Endpoint 7 Buffer Address Register
diepctl8 on page 18-910	0x200	32	RW	0x0	Device Control IN Endpoint 8 Control Register
diepint8 on page 18-919	0x208	32	RW	0x80	Device IN Endpoint 8 Interrupt Register
dieptsiz8 on page 18-926	0x210	32	RW	0x0	Device IN Endpoint 8 Transfer Size Register

Register	Offset	Width	Access	Reset Value	Description
diepdma8 on page 18-928	0x214	32	RW	0x0	Device IN Endpoint 8 DMA Address Register
dtxfsts8 on page 18-929	0x218	32	RO	0x2000	Device IN Endpoint Transmit FIFO Status Register 8
diepdmab8 on page 18-930	0x21C	32	RO	0x0	Device IN Endpoint 8 Buffer Address Register
diepctl9 on page 18-931	0x220	32	RW	0x0	Device Control IN Endpoint 9 Control Register
diepint9 on page 18-940	0x228	32	RW	0x80	Device IN Endpoint 9 Interrupt Register
dieptsiz9 on page 18-947	0x230	32	RW	0x0	Device IN Endpoint 9 Transfer Size Register
diepdma9 on page 18-949	0x234	32	RW	0x0	Device IN Endpoint 9 DMA Address Register
dtxfsts9 on page 18-950	0x238	32	RO	0x2000	Device IN Endpoint Transmit FIFO Status Register 9
diepdmab9 on page 18-951	0x23C	32	RO	0x0	Device IN Endpoint 9 Buffer Address Register
diepctl10 on page 18-952	0x240	32	RW	0x0	Device Control IN Endpoint 10 Control Register
diepint10 on page 18-961	0x248	32	RW	0x80	Device IN Endpoint 10 Interrupt Register
dieptsiz10 on page 18-968	0x250	32	RW	0x0	Device IN Endpoint 10 Transfer Size Register
diepdma10 on page 18-970	0x254	32	RW	0x0	Device IN Endpoint 10 DMA Address Register
dtxfsts10 on page 18-971	0x258	32	RO	0x2000	Device IN Endpoint Transmit FIFO Status Register 10
diepdmab10 on page 18-972	0x25C	32	RO	0x0	Device IN Endpoint 10 Buffer Address Register

Register	Offset	Width	Access	Reset Value	Description
diepctl11 on page 18-973	0x260	32	RW	0x0	Device Control IN Endpoint 11 Control Register
diepint11 on page 18-982	0x268	32	RW	0x80	Device IN Endpoint 11 Interrupt Register
dieptsiz11 on page 18-989	0x270	32	RW	0x0	Device IN Endpoint 11 Transfer Size Register
diepdma11 on page 18-991	0x274	32	RW	0x0	Device IN Endpoint 11 DMA Address Register
dtxfsts11 on page 18-992	0x278	32	RO	0x2000	Device IN Endpoint Transmit FIFO Status Register 11
diepdmab11 on page 18-993	0x27C	32	RO	0x0	Device IN Endpoint 11 Buffer Address Register
diepctl12 on page 18-994	0x280	32	RW	0x0	Device Control IN Endpoint 12 Control Register
diepint12 on page 18-1003	0x288	32	RW	0x80	Device IN Endpoint 12 Interrupt Register
dieptsiz12 on page 18-1010	0x290	32	RW	0x0	Device IN Endpoint 12 Transfer Size Register
diepdma12 on page 18-1012	0x294	32	RW	0x0	Device IN Endpoint 12 DMA Address Register
dtxfsts12 on page 18-1013	0x298	32	RO	0x2000	Device IN Endpoint Transmit FIFO Status Register 12
diepdmab12 on page 18-1014	0x29C	32	RO	0x0	Device IN Endpoint 12 Buffer Address Register
diepctl13 on page 18-1015	0x2A0	32	RW	0x0	Device Control IN Endpoint 13 Control Register
diepint13 on page 18-1024	0x2A8	32	RW	0x80	Device IN Endpoint 13 Interrupt Register
dieptsiz13 on page 18-1031	0x2B0	32	RW	0x0	Device IN Endpoint 13 Transfer Size Register

Register	Offset	Width	Access	Reset Value	Description
diepdma13 on page 18-1033	0x2B4	32	RW	0x0	Device IN Endpoint 13 DMA Address Register
dtxfsts13 on page 18-1034	0x2B8	32	RO	0x2000	Device IN Endpoint Transmit FIFO Status Register 13
diepdmab13 on page 18-1035	0x2BC	32	RO	0x0	Device IN Endpoint 13 Buffer Address Register
diepctl14 on page 18-1036	0x2C0	32	RW	0x0	Device Control IN Endpoint 14 Control Register
diepint14 on page 18-1045	0x2C8	32	RW	0x80	Device IN Endpoint 14 Interrupt Register
dieptsiz14 on page 18-1052	0x2D0	32	RW	0x0	Device IN Endpoint 14 Transfer Size Register
diepdma14 on page 18-1054	0x2D4	32	RW	0x0	Device IN Endpoint 14 DMA Address Register
dtxfsts14 on page 18-1055	0x2D8	32	RO	0x2000	Device IN Endpoint Transmit FIFO Status Register 14
diepdmab14 on page 18-1056	0x2DC	32	RO	0x0	Device IN Endpoint 14 Buffer Address Register
diepctl15 on page 18-1057	0x2E0	32	RW	0x0	Device Control IN Endpoint 15 Control Register
diepint15 on page 18-1066	0x2E8	32	RW	0x80	Device IN Endpoint 15 Interrupt Register
dieptsiz15 on page 18-1073	0x2F0	32	RW	0x0	Device IN Endpoint 15 Transfer Size Register
diepdma15 on page 18-1075	0x2F4	32	RW	0x0	Device IN Endpoint 15 DMA Address Register
dtxfsts15 on page 18-1076	0x2F8	32	RO	0x2000	Device IN Endpoint Transmit FIFO Status Register 15
diepdmab15 on page 18-1077	0x2FC	32	RO	0x0	Device IN Endpoint 15 Buffer Address Register

Register	Offset	Width	Access	Reset Value	Description
doepctl0 on page 18-1078	0x300	32	RW	0x8000	Device Control OUT Endpoint 0 Control Register
doepint0 on page 18-1083	0x308	32	RW	0x0	Device OUT Endpoint 0 Interrupt Register
doeptsiz0 on page 18-1089	0x310	32	RW	0x0	Device OUT Endpoint 0 Transfer Size Register
doepdma0 on page 18-1091	0x314	32	RW	0x0	Device OUT Endpoint 0 DMA Address Register
doepdmab0 on page 18-1092	0x31C	32	RO	0x0	Device OUT Endpoint 16 Buffer Address Register
doepctl1 on page 18-1093	0x320	32	RW	0x0	Device Control OUT Endpoint 1 Control Register
doepint1 on page 18-1102	0x328	32	RW	0x0	Device OUT Endpoint 1 Interrupt Register
doeptsiz1 on page 18-1109	0x330	32	RW	0x0	Device OUT Endpoint 1 Transfer Size Register
doepdma1 on page 18-1111	0x334	32	RW	0x0	Device OUT Endpoint 1 DMA Address Register
doepdmab1 on page 18-1112	0x33C	32	RO	0x0	Device OUT Endpoint 1 Buffer Address Register
doepctl2 on page 18-1113	0x340	32	RW	0x0	Device Control OUT Endpoint 2 Control Register
doepint2 on page 18-1122	0x348	32	RW	0x0	Device OUT Endpoint 2 Interrupt Register
doeptsiz2 on page 18-1129	0x350	32	RW	0x0	Device OUT Endpoint 2 Transfer Size Register
doepdma2 on page 18-1131	0x354	32	RW	0x0	Device OUT Endpoint 2 DMA Address Register
doepdmab2 on page 18-1132	0x35C	32	RO	0x0	Device OUT Endpoint 2 Buffer Address Register

Register	Offset	Width	Access	Reset Value	Description
doepctl3 on page 18-1133	0x360	32	RW	0x0	Device Control OUT Endpoint 3 Control Register
doepint3 on page 18-1142	0x368	32	RW	0x0	Device OUT Endpoint 3 Interrupt Register
doeptsiz3 on page 18-1149	0x370	32	RW	0x0	Device OUT Endpoint 3 Transfer Size Register
doepdma3 on page 18-1151	0x374	32	RW	0x0	Device OUT Endpoint 3 DMA Address Register
doepdmab3 on page 18-1152	0x37C	32	RO	0x0	Device OUT Endpoint 3 Buffer Address Register
doepctl4 on page 18-1153	0x380	32	RW	0x0	Device Control OUT Endpoint 4 Control Register
doepint4 on page 18-1162	0x388	32	RW	0x0	Device OUT Endpoint 4 Interrupt Register
doeptsiz4 on page 18-1169	0x390	32	RW	0x0	Device OUT Endpoint 4 Transfer Size Register
doepdma4 on page 18-1171	0x394	32	RW	0x0	Device OUT Endpoint 4 DMA Address Register
doepdmab4 on page 18-1172	0x39C	32	RO	0x0	Device OUT Endpoint 4 Buffer Address Register
doepctl5 on page 18-1173	0x3A0	32	RW	0x0	Device Control OUT Endpoint 5 Control Register
doepint5 on page 18-1182	0x3A8	32	RW	0x0	Device OUT Endpoint 5 Interrupt Register
doeptsiz5 on page 18-1189	0x3B0	32	RW	0x0	Device OUT Endpoint 5 Transfer Size Register
doepdma5 on page 18-1191	0x3B4	32	RW	0x0	Device OUT Endpoint 5 DMA Address Register
doepdmab5 on page 18-1192	0x3BC	32	RO	0x0	Device OUT Endpoint 5 Buffer Address Register

Register	Offset	Width	Access	Reset Value	Description
doepctl6 on page 18-1193	0x3C0	32	RW	0x0	Device Control OUT Endpoint 6 Control Register
doepint6 on page 18-1202	0x3C8	32	RW	0x0	Device OUT Endpoint 6 Interrupt Register
doeptsiz6 on page 18-1209	0x3D0	32	RW	0x0	Device OUT Endpoint 6 Transfer Size Register
doepdma6 on page 18-1211	0x3D4	32	RW	0x0	Device OUT Endpoint 6 DMA Address Register
doepdmab6 on page 18-1212	0x3DC	32	RO	0x0	Device OUT Endpoint 6 Buffer Address Register
doepctl7 on page 18-1213	0x3E0	32	RW	0x0	Device Control OUT Endpoint 7 Control Register
doepint7 on page 18-1222	0x3E8	32	RW	0x0	Device OUT Endpoint 7 Interrupt Register
doeptsiz7 on page 18-1229	0x3F0	32	RW	0x0	Device OUT Endpoint 7 Transfer Size Register
doepdma7 on page 18-1231	0x3F4	32	RW	0x0	Device OUT Endpoint 7 DMA Address Register
doepdmab7 on page 18-1232	0x3FC	32	RO	0x0	Device OUT Endpoint 7 Buffer Address Register
doepctl8 on page 18-1233	0x400	32	RW	0x0	Device Control OUT Endpoint 8 Control Register
doepint8 on page 18-1242	0x408	32	RW	0x0	Device OUT Endpoint 8 Interrupt Register
doeptsiz8 on page 18-1249	0x410	32	RW	0x0	Device OUT Endpoint 8 Transfer Size Register
doepdma8 on page 18-1251	0x414	32	RW	0x0	Device OUT Endpoint 8 DMA Address Register
doepdmab8 on page 18-1252	0x41C	32	RO	0x0	Device OUT Endpoint 8 Buffer Address Register

Register	Offset	Width	Access	Reset Value	Description
doepctl9 on page 18-1253	0x420	32	RW	0x0	Device Control OUT Endpoint 9 Control Register
doepint9 on page 18-1262	0x428	32	RW	0x0	Device OUT Endpoint 9 Interrupt Register
doeptsiz9 on page 18-1269	0x430	32	RW	0x0	Device OUT Endpoint 9 Transfer Size Register
doepdma9 on page 18-1271	0x434	32	RW	0x0	Device OUT Endpoint 9 DMA Address Register
doepdmab9 on page 18-1272	0x43C	32	RO	0x0	Device OUT Endpoint 9 Buffer Address Register
doepctl10 on page 18-1273	0x440	32	RW	0x0	Device Control OUT Endpoint 10 Control Register
doepint10 on page 18-1282	0x448	32	RW	0x0	Device OUT Endpoint 10 Interrupt Register
doeptsiz10 on page 18-1289	0x450	32	RW	0x0	Device OUT Endpoint 10 Transfer Size Register
doepdma10 on page 18-1291	0x454	32	RW	0x0	Device OUT Endpoint 10 DMA Address Register
doepdmab10 on page 18-1292	0x45C	32	RO	0x0	Device OUT Endpoint 10 Buffer Address Register
doepctl11 on page 18-1293	0x460	32	RW	0x0	Device Control OUT Endpoint 11 Control Register
doepint11 on page 18-1302	0x468	32	RW	0x0	Device OUT Endpoint 11 Interrupt Register
doeptsiz11 on page 18-1309	0x470	32	RW	0x0	Device OUT Endpoint 11 Transfer Size Register
doepdma11 on page 18-1311	0x474	32	RW	0x0	Device OUT Endpoint 11 DMA Address Register
doepdmab11 on page 18-1312	0x47C	32	RO	0x0	Device OUT Endpoint 11 Buffer Address Register

Register	Offset	Width	Access	Reset Value	Description
doepctl12 on page 18-1313	0x480	32	RW	0x0	Device Control OUT Endpoint 12 Control Register
doepint12 on page 18-1322	0x488	32	RW	0x0	Device OUT Endpoint 12 Interrupt Register
doeptsiz12 on page 18-1329	0x490	32	RW	0x0	Device OUT Endpoint 12 Transfer Size Register
doepdma12 on page 18-1331	0x494	32	RW	0x0	Device OUT Endpoint 12 DMA Address Register
doepdmab12 on page 18-1332	0x49C	32	RO	0x0	Device OUT Endpoint 12 Buffer Address Register
doepctl13 on page 18-1333	0x4A0	32	RW	0x0	Device Control OUT Endpoint 13 Control Register
doepint13 on page 18-1342	0x4A8	32	RW	0x0	Device OUT Endpoint 13 Interrupt Register
doeptsiz13 on page 18-1349	0x4B0	32	RW	0x0	Device OUT Endpoint 13 Transfer Size Register
doepdma13 on page 18-1351	0x4B4	32	RW	0x0	Device OUT Endpoint 13 DMA Address Register
doepdmab13 on page 18-1352	0x4BC	32	RO	0x0	Device OUT Endpoint 13 Buffer Address Register
doepctl14 on page 18-1353	0x4C0	32	RW	0x0	Device Control OUT Endpoint 14 Control Register
doepint14 on page 18-1362	0x4C8	32	RW	0x0	Device OUT Endpoint 14 Interrupt Register
doeptsiz14 on page 18-1369	0x4D0	32	RW	0x0	Device OUT Endpoint 14 Transfer Size Register
doepdma14 on page 18-1371	0x4D4	32	RW	0x0	Device OUT Endpoint 14 DMA Address Register
doepdmab14 on page 18-1372	0x4DC	32	RO	0x0	Device OUT Endpoint 14 Buffer Address Register

Register	Offset	Width	Access	Reset Value	Description
doepctl15 on page 18-1373	0x4E0	32	RW	0x0	Device Control OUT Endpoint 15 Control Register
doepint15 on page 18-1382	0x4E8	32	RW	0x0	Device OUT Endpoint 15 Interrupt Register
doeptsiz15 on page 18-1389	0x4F0	32	RW	0x0	Device OUT Endpoint 15 Transfer Size Register
doepdma15 on page 18-1391	0x4F4	32	RW	0x0	Device OUT Endpoint 15 DMA Address Register
doepdmab15 on page 18-1392	0x4FC	32	RO	0x0	Device OUT Endpoint 15 Buffer Address Register

usb_devgrp Summary

Module Instance	Base Address
i_usbotg_0_devgrp	0xFFB00800
i_usbotg_1_devgrp	0xFFB40800

Register Address Offset	Bit Fields															
i_usbotg_0_devgrp																
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	resvalid RW 0x2						perschintvl RW 0x0		descdma RW 0x0	Reserved						
dcfg 0x0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	errticintmsk RW 0x0	xcvrdly RW 0x0	endevoutnak RW 0x0	perfint RW 0x0		devaddr RW 0x0							ena32khzsusp RW 0x0	nzstsouthshk RW 0x0	devspd RW 0x0	

Register Address Offset	Bit Fields															
dctl 0x4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved														enconton bna RW 0x0	nakonbble RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ignrfrmnum RW 0x0	gmc RW 0x0		Reserved	pwronprgdone RW 0x0	cgoutnak WO 0x0	sgoutnak WO 0x0	cgnpinnak WO 0x0	sgnpinnak WO 0x0	tstctl RW 0x0			goutnaksts RO 0x0	gnpinnaksts RO 0x0	sftdiscon RW 0x1	rmtwkupsig RW 0x0
dsts 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved								devlnsts RO 0x0		soffn RO 0x0					
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	soffn RO 0x0								Reserved				errticerr RO 0x0	enumspd RO 0x1		suspsts RO 0x0
diepmsk 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		nakmsk RW 0x0	Reserved			bnaintrmsk RW 0x0	txfifoundrnmsk RW 0x0	Reserved	inepnakeffmsk RW 0x0	intknepmismsk RW 0x0	intkntxfempmsk RW 0x0	timeoutmsk RW 0x0	ahberrmsk RW 0x0	epdisbldmsk RW 0x0	xfercomplmsk RW 0x0
doepmsk 0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	nyetmsk RW 0x0	nakmsk RW 0x0	bbleerrmsk RW 0x0	Reserved		bnaoutintrmsk RW 0x0	outpkterrmsk RW 0x0	Reserved	back2backsetup RW 0x0	stsphsercvdmsk RW 0x0	outtknepdismsk RW 0x0	setupmsk RW 0x0	ahberrmsk RW 0x0	epdisbldmsk RW 0x0	xfercomplmsk RW 0x0

Register Address Offset	Bit Fields															
daint 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	outepint15 RO 0x0	outepint14 RO 0x0	outepint13 RO 0x0	outepint12 RO 0x0	outepint11 RO 0x0	outepint10 RO 0x0	outepint9 RO 0x0	outepint8 RO 0x0	outepint7 RO 0x0	outepint6 RO 0x0	outepint5 RO 0x0	outepint4 RO 0x0	outepint3 RO 0x0	outepint2 RO 0x0	outepint1 RO 0x0	outepint0 RO 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	inepint15 RO 0x0	inepint14 RO 0x0	inepint13 RO 0x0	inepint12 RO 0x0	inepint11 RO 0x0	inepint10 RO 0x0	inepint9 RO 0x0	inepint8 RO 0x0	inepint7 RO 0x0	inepint6 RO 0x0	inepint5 RO 0x0	inepint4 RO 0x0	inepint3 RO 0x0	inepint2 RO 0x0	inepint1 RO 0x0	inepint0 RO 0x0
daintmsk 0x1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	outepmsk15 RW 0x0	outepmsk14 RW 0x0	outepmsk13 RW 0x0	outepmsk12 RW 0x0	outepmsk11 RW 0x0	outepmsk10 RW 0x0	outepmsk9 RW 0x0	outepmsk8 RW 0x0	outepmsk7 RW 0x0	outepmsk6 RW 0x0	outepmsk5 RW 0x0	outepmsk4 RW 0x0	outepmsk3 RW 0x0	outepmsk2 RW 0x0	outepmsk1 RW 0x0	outepmsk0 RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	inepmsk15 RW 0x0	inepmsk14 RW 0x0	inepmsk13 RW 0x0	inepmsk12 RW 0x0	inepmsk11 RW 0x0	inepmsk10 RW 0x0	inepmsk9 RW 0x0	inepmsk8 RW 0x0	inepmsk7 RW 0x0	inepmsk6 RW 0x0	inepmsk5 RW 0x0	inepmsk4 RW 0x0	inepmsk3 RW 0x0	inepmsk2 RW 0x0	inepmsk1 RW 0x0	inepmsk0 RW 0x0
dvbusdis 0x28	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	dvbusdis RW 0x17D7															
dvbuspulse 0x2C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				dvbuspulse RW 0x5B8											
dthrctl 0x30	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved				arbprken RW 0x1	Reserved	rxthrlen RW 0x8									rxthren RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved			ahbthrratio RW 0x0		txthrlen RW 0x8									isothren RW 0x0	nonisothren RW 0x0

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
diepempmsk 0x34	inepTxfEmpMsk15 RW 0x0	inepTxfEmpMsk14 RW 0x0	inepTxfEmpMsk13 RW 0x0	inepTxfEmpMsk12 RW 0x0	inepTxfEmpMsk11 RW 0x0	inepTxfEmpMsk10 RW 0x0	inepTxfEmpMsk9 RW 0x0	inepTxfEmpMsk8 RW 0x0	inepTxfEmpMsk7 RW 0x0	inepTxfEmpMsk6 RW 0x0	inepTxfEmpMsk5 RW 0x0	inepTxfEmpMsk4 RW 0x0	inepTxfEmpMsk3 RW 0x0	inepTxfEmpMsk2 RW 0x0	inepTxfEmpMsk1 RW 0x0	ineptxfempmsk0 RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	epena RW 0x0	epdis RW 0x0	Reserved		snak WO 0x0	cnak WO 0x0	txfnum RW 0x0				stall RW 0x0	Reserved	eptype RO 0x0		naksts RO 0x0	Reserved
diepctl0 0x100	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	usbactep RO 0x1	Reserved													mps RW 0x0	
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
diepint0 0x108	Reserved	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	txfifoundrn RW 0x0	txfemp RO 0x1	inepnakeff RW 0x0	intknepmis RW 0x0	intkntxfemp RW 0x0	timeout RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved											pktcnt RW 0x0		Reserved		
dieptsiz0 0x110	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved									xfersize RW 0x0						
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	diepdma0 RW 0x0															
diepdma0 0x114	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	diepdma0 RW 0x0															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dtxfsts0 0x118	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ineptxfspcavail RO 0x2000															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepdmab0 0x11C	diepdmab0 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	diepdmab0 RO 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepctl1 0x120	epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	txfnum RW 0x0				stall RW 0x0	Reserved	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	usbactep RW 0x0	Reserved				mps RW 0x0										
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepint1 0x128	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	txfifoundrn RW 0x0	txfemp RO 0x1	inepnakeff RW 0x0	intknepmis RW 0x0	intkntxfemp RW 0x0	timeout RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dieptsiz1 0x130	Reserved	mc RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepdma1 0x134	diepdma1 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	diepdma1 RW 0x0															
dtxfsts1 0x138	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ineptxfspcavail RO 0x2000															
diepdmab1 0x13C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	diepdmab1 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	diepdmab1 RO 0x0															
diepctl2 0x140	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	txfnum RW 0x0				stall RW 0x0	Reserved	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	usbactep RW 0x0	Reserved				mps RW 0x0										
diepint2 0x148	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	txfifoundrn RW 0x0	txfemp RO 0x1	inepnakeff RW 0x0	intknepmis RW 0x0	intkntxfemp RW 0x0	timeout RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0

Register Address Offset	Bit Fields															
dieptsiz2 0x150	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	mc RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
diepdma2 0x154	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	diepdma2 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	diepdma2 RW 0x0															
dtxfsts2 0x158	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ineptxfspcavail RO 0x2000															
diepdmab2 0x15C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	diepdmab2 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	diepdmab2 RO 0x0															
diepctl3 0x160	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	txfnum RW 0x0				stall RW 0x0	Reserved	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	usbactep RW 0x0	Reserved				mps RW 0x0										

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
diepint3 0x168	Reserved	nyetintrpt RW 0x0	nakintrpt RW 0x0	bleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	txfifoundrn RW 0x0	txfemp RO 0x1	inepnakeff RW 0x0	intknepmis RW 0x0	intkntxfemp RW 0x0	timeout RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	mc RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
dieptsiz3 0x170	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	diepdma3 RW 0x0															
diepdma3 0x174	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	diepdma3 RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
dtxfsts3 0x178	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ineptxfspcavail RO 0x2000															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	diepdmab3 RO 0x0															
diepdmab3 0x17C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	diepdmab3 RO 0x0															

Register Address Offset	Bit Fields															
diepctl4 0x180	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	txfnum RW 0x0				stall RW 0x0	Reserved	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	usbactep RW 0x0	Reserved				mps RW 0x0										
diepint4 0x188	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	nyetintrpt RW 0x0	nakintrpt RW 0x0	bleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	txfifoundrn RW 0x0	txfemp RO 0x1	inepnakeff RW 0x0	intknepmis RW 0x0	intkntxfemp RW 0x0	timeout RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0
dieptsiz4 0x190	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	mc RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
diepdma4 0x194	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	diepdma4 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	diepdma4 RW 0x0															
dtxfsts4 0x198	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ineptxfspcavail RO 0x2000															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepdmab4	diepdmab4 RO 0x0															
0x19C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	diepdmab4 RO 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepctl5	epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	txfnum RW 0x0				stall RW 0x0	Reserved	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
0x1A0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	usbactep RW 0x0	Reserved				mps RW 0x0										
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
diepint5	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x1A8	Reserved	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	txfifoundrn RW 0x0	txfemp RO 0x1	inepnakeff RW 0x0	intknepmis RW 0x0	intkntxfemp RW 0x0	timeout RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dieptsiz5	Reserved	mc RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
0x1B0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepdma5	diepdma5 RW 0x0															
0x1B4	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	diepdma5 RW 0x0															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dtxfsts5	Reserved															
0x1B8	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ineptxfspcavail RO 0x2000															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepdmab5	diepdmab5 RO 0x0															
0x1BC	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	diepdmab5 RO 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepctl6	epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	txfnum RW 0x0				stall RW 0x0	Reserved	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
0x1C0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	usbactep RW 0x0	Reserved				mps RW 0x0										
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepint6	Reserved															
0x1C8	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	nyetintrpt RW 0x0	nakintrpt RW 0x0	bleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	txfifoundrn RW 0x0	txfemp RO 0x1	inepnakeff RW 0x0	intknepmis RW 0x0	intkntxfemp RW 0x0	timeout RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dieptsiz6	Reserved	mc RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
0x1D0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepdma6 0x1D4	diepdma6 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	diepdma6 RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dtxfsts6 0x1D8	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ineptxfspcavail RO 0x2000															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepdmab6 0x1DC	diepdmab6 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	diepdmab6 RO 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepctl7 0x1E0	epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	txfnum RW 0x0				stall RW 0x0	Reserved	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	usbactep RW 0x0	Reserved				mps RW 0x0										
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepint7 0x1E8	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	txfifoundrn RW 0x0	txfemp RO 0x1	inepnakeff RW 0x0	intknepmis RW 0x0	intkntxfemp RW 0x0	timeout RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	mc RW 0x0			pktcnt RW 0x0									xfersize RW 0x0		
dieptsiz7 0x1F0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	diepdma7 RW 0x0															
diepdma7 0x1F4	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	diepdma7 RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
dtxfsts7 0x1F8	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ineptxfspcavail RO 0x2000															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	diepdmab7 RO 0x0															
diepdmab7 0x1FC	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	diepdmab7 RO 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	txfnum RW 0x0				stall RW 0x0	Reserved	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
diepctl8 0x200	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	usbactep RW 0x0	Reserved				mps RW 0x0										

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
diepint8 0x208	Reserved	nyetintrpt RW 0x0	nakintrpt RW 0x0	bleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	txfifoundrn RW 0x0	txfemp RO 0x1	inepnakeff RW 0x0	intknepmis RW 0x0	intkntxfemp RW 0x0	timeout RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	mc RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
dieptsiz8 0x210	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	diepdma8 RW 0x0															
diepdma8 0x214	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	diepdma8 RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
dtxfsts8 0x218	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ineptxfspcavail RO 0x2000															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	diepdmab8 RO 0x0															
diepdmab8 0x21C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	diepdmab8 RO 0x0															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	txfnum RW 0x0				stall RW 0x0	Reserved	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
diepctl9 0x220	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	usbactep RW 0x0	Reserved				mps RW 0x0										
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
diepint9 0x228	Reserved	nyetintrpt RW 0x0	nakintrpt RW 0x0	bleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	txfifoundrn RW 0x0	txfemp RO 0x1	inepnakeff RW 0x0	intknepmis RW 0x0	intkntxfemp RW 0x0	timeout RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	mc RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
dieptsiz9 0x230	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	diepdma9 RW 0x0															
diepdma9 0x234	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	diepdma9 RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
dtxfsts9 0x238	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ineptxfspcavail RO 0x2000															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepdmab9	diepdmab9 RO 0x0															
0x23C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	diepdmab9 RO 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepctl10	epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	txfnum RW 0x0				stall RW 0x0	Reserved	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
0x240	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	usbactep RW 0x0	Reserved				mps RW 0x0										
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
diepint10	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x248	Reserved	nyetintrpt RW 0x0	nakintrpt RW 0x0	bleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	txfifoundrn RW 0x0	txfemp RO 0x1	inepnakeff RW 0x0	intknepmis RW 0x0	intkntxfemp RW 0x0	timeout RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dieptsiz10	Reserved	mc RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
0x250	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepdma10	diepdma10 RW 0x0															
0x254	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	diepdma10 RW 0x0															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dtxfsts10	Reserved															
0x258	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ineptxfspcavail RO 0x2000															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepdmab10	diepdmab10 RO 0x0															
0x25C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	diepdmab10 RO 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepctl11	epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	txfnum RW 0x0				stall RW 0x0	Reserved	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
0x260	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	usbactep RW 0x0	Reserved				mps RW 0x0										
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepint11	Reserved															
0x268	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	nyetintrpt RW 0x0	nakintrpt RW 0x0	bleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	txfifoundrn RW 0x0	txfemp RO 0x1	inepnakeff RW 0x0	intknepmis RW 0x0	intkntxfemp RW 0x0	timeout RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dieptsiz11	Reserved	mc RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
0x270	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepdma11 0x274	diepdma11 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	diepdma11 RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dtxfsts11 0x278	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ineptxfspcavail RO 0x2000															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepdmab11 0x27C	diepdmab11 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	diepdmab11 RO 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepctl12 0x280	epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	txfnum RW 0x0				stall RW 0x0	Reserved	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	usbactep RW 0x0	Reserved				mps RW 0x0										
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepint12 0x288	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	txfifoundrn RW 0x0	txfemp RO 0x1	inepnakeff RW 0x0	intknepmis RW 0x0	intkntxfemp RW 0x0	timeout RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	mc RW 0x0			pktcnt RW 0x0									xfersize RW 0x0		
dieptsiz12 0x290	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	diepdma12 RW 0x0															
diepdma12 0x294	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	diepdma12 RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
dtxfsts12 0x298	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ineptxfspcavail RO 0x2000															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	diepdmab12 RO 0x0															
diepdmab12 0x29C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	diepdmab12 RO 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	txfnum RW 0x0				stall RW 0x0	Reserved	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
diepctl13 0x2A0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	usbactep RW 0x0	Reserved				mps RW 0x0										

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
diepint13 0x2A8	Reserved	nyetintrpt RW 0x0	nakintrpt RW 0x0	bleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	txfifoundrn RW 0x0	txfemp RO 0x1	inepnakeff RW 0x0	intknepmis RW 0x0	intkntxfemp RW 0x0	timeout RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	mc RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
dieptsiz13 0x2B0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	diepdma13 RW 0x0															
diepdma13 0x2B4	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	diepdma13 RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
dtxfsts13 0x2B8	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ineptxfspcavail RO 0x2000															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	diepdmab13 RO 0x0															
diepdmab13 0x2BC	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	diepdmab13 RO 0x0															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	txfnum RW 0x0				stall RW 0x0	Reserved	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
diepctl14 0x2C0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	usbactep RW 0x0	Reserved				mps RW 0x0										
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
diepint14 0x2C8	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	txfifoundrn RW 0x0	txfemp RO 0x1	inepnakeff RW 0x0	intknepmis RW 0x0	intkntxfemp RW 0x0	timeout RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	mc RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
dieptsiz14 0x2D0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	diepdma14 RW 0x0															
diepdma14 0x2D4	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	diepdma14 RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
dtxfsts14 0x2D8	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ineptxfspcavail RO 0x2000															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepdmab14	diepdmab14 RO 0x0															
0x2DC	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	diepdmab14 RO 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepctl15	epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	txfnum RW 0x0				stall RW 0x0	Reserved	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
0x2E0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	usbactep RW 0x0	Reserved				mps RW 0x0										
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
diepint15	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x2E8	Reserved	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	txfifoundrn RW 0x0	txfemp RO 0x1	inepnakeff RW 0x0	intknepmis RW 0x0	intkntxfemp RW 0x0	timeout RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dieptsiz15	Reserved	mc RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
0x2F0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepdma15	diepdma15 RW 0x0															
0x2F4	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	diepdma15 RW 0x0															

Register Address Offset	Bit Fields															
dtxfsts15 0x2F8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ineptxfspcavail RO 0x2000															
diepdmab15 0x2FC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	diepdmab15 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	diepdmab15 RO 0x0															
doepctl0 0x300	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	epena RW 0x0	epdis RO 0x0	Reserved		snak WO 0x0	cnak WO 0x0	Reserved				stall RW 0x0	snp RW 0x0	eptype RO 0x0		naksts RO 0x0	Reserved
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	usbactep RO 0x1	Reserved													mps RO 0x0	
doepint0 0x308	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	stuppktrcvd RW 0x0	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	outpkterr RW 0x0	Reserved	back2backsetup RW 0x0	stsphsercvd RW 0x0	outtknepdis RW 0x0	setup RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0
doeptsiz0 0x310	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	supcnt RW 0x0		Reserved									pktcnt RW 0x0	Reserved		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved									xfersize RW 0x0						

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdma0	doepdma0 RW 0x0															
0x314	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	doepdma0 RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdmab0	doepdmab0 RO 0x0															
0x31C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	doepdmab0 RO 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepctl1	epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	Reserved				stall RW 0x0	snp RW 0x0	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
0x320	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	usbactep RW 0x0	Reserved				mps RW 0x0										
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepint1	Reserved															
0x328	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	stuppktrcvd RW 0x0	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	outpkterr RW 0x0	Reserved	back2backsetup RW 0x0	stsphsercvd RW 0x0	outtknepdis RW 0x0	setup RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doeptsiz1	Reserved	rxdpid RO 0x0		pktcnt RW 0x0										xfersize RW 0x0		
0x330	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdma1	doepdma1 RW 0x0															
0x334	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	doepdma1 RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdmab1	doepdmab1 RO 0x0															
0x33C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	doepdmab1 RO 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepctl2	epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	Reserved				stall RW 0x0	snp RW 0x0	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
0x340	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	usbactep RW 0x0	Reserved				mps RW 0x0										
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepint2	Reserved															
0x348	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	stuppktrcvd RW 0x0	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	outpkterr RW 0x0	Reserved	back2backsetup RW 0x0	stsphsercvd RW 0x0	outtknepdis RW 0x0	setup RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doeptsiz2	Reserved	rxdpid RO 0x0		pktcnt RW 0x0										xfersize RW 0x0		
0x350	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdma2 0x354	doepdma2 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	doepdma2 RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdmab2 0x35C	doepdmab2 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	doepdmab2 RO 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepctl3 0x360	epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	Reserved				stall RW 0x0	snp RW 0x0	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	usbactep RW 0x0	Reserved				mps RW 0x0										
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepint3 0x368	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	stuppktrcvd RW 0x0	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	outpkterr RW 0x0	Reserved	back2backsetup RW 0x0	stsphsercvd RW 0x0	outtknepdis RW 0x0	setup RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doeptsiz3 0x370	Reserved	rxdpid RO 0x0		pktcnt RW 0x0										xfersize RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdma3	doepdma3 RW 0x0															
0x374	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	doepdma3 RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdmab3	doepdmab3 RO 0x0															
0x37C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	doepdmab3 RO 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepctl4	epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	Reserved				stall RW 0x0	snp RW 0x0	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
0x380	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	usbactep RW 0x0	Reserved				mps RW 0x0										
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepint4	Reserved															
0x388	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	stuppktrcvd RW 0x0	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	outpkterr RW 0x0	Reserved	back2backsetup RW 0x0	stsphsercvd RW 0x0	outtknepdis RW 0x0	setup RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doeptsiz4	Reserved	rxdpid RO 0x0		pktcnt RW 0x0										xfersize RW 0x0		
0x390	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdma4 0x394	doepdma4 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	doepdma4 RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdmab4 0x39C	doepdmab4 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	doepdmab4 RO 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepctl5 0x3A0	epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	Reserved				stall RW 0x0	snp RW 0x0	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	usbactep RW 0x0	Reserved				mps RW 0x0										
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepint5 0x3A8	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	stuppktrcvd RW 0x0	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	outpkterr RW 0x0	Reserved	back2backsetup RW 0x0	stsphsercvd RW 0x0	outtknepdis RW 0x0	setup RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doeptsiz5 0x3B0	Reserved	rxdpid RO 0x0		pktcnt RW 0x0										xfersize RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															

Register Address Offset	Bit Fields															
doepdma5 0x3B4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	doepdma5 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	doepdma5 RW 0x0															
doepdmab5 0x3BC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	doepdmab5 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	doepdmab5 RO 0x0															
doepctl6 0x3C0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	Reserved				stall RW 0x0	snp RW 0x0	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	usbactep RW 0x0	Reserved				mps RW 0x0										
doepint6 0x3C8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	stuppktrcvd RW 0x0	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	outpkterr RW 0x0	Reserved	back2backsetup RW 0x0	stsphsercvd RW 0x0	outtknepdis RW 0x0	setup RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0
doeptsiz6 0x3D0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	rxdpid RO 0x0		pktcnt RW 0x0										xfersize RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdma6 0x3D4	doepdma6 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	doepdma6 RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdmab6 0x3DC	doepdmab6 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	doepdmab6 RO 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepctl7 0x3E0	epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	Reserved				stall RW 0x0	snp RW 0x0	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	usbactep RW 0x0	Reserved				mps RW 0x0										
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
doepint7 0x3E8	stuppktrcvd RW 0x0	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	outpkterr RW 0x0	Reserved	back2backsetup RW 0x0	stsphsercvd RW 0x0	outtknepdis RW 0x0	setup RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doeptsiz7 0x3F0	Reserved	rxdpid RO 0x0		pktcnt RW 0x0										xfersize RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdma7 0x3F4	doepdma7 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	doepdma7 RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdmab7 0x3FC	doepdmab7 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	doepdmab7 RO 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepctl8 0x400	epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	Reserved				stall RW 0x0	snp RW 0x0	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	usbactep RW 0x0	Reserved				mps RW 0x0										
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepint8 0x408	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	stuppktrcvd RW 0x0	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	outpkterr RW 0x0	Reserved	back2backsetup RW 0x0	stsphsercvd RW 0x0	outtknepdis RW 0x0	setup RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doeptsiz8 0x410	Reserved	rxdpid RO 0x0		pktcnt RW 0x0										xfersize RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdma8 0x414	doepdma8 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	doepdma8 RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdmab8 0x41C	doepdmab8 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	doepdmab8 RO 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepctl9 0x420	epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	Reserved				stall RW 0x0	snp RW 0x0	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	usbactep RW 0x0	Reserved				mps RW 0x0										
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepint9 0x428	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	stuppktrcvd RW 0x0	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	outpkterr RW 0x0	Reserved	back2backsetup RW 0x0	stsphsercvd RW 0x0	outtknepdis RW 0x0	setup RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doeptsiz9 0x430	Reserved	rxdpid RO 0x0		pktcnt RW 0x0										xfersize RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdma9 0x434	doepdma9 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	doepdma9 RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdmab9 0x43C	doepdmab9 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	doepdmab9 RO 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepctl10 0x440	epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	Reserved				stall RW 0x0	snp RW 0x0	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	usbactep RW 0x0	Reserved				mps RW 0x0										
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepint10 0x448	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	stuppktrcvd RW 0x0	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	outpkterr RW 0x0	Reserved	back2backsetup RW 0x0	stsphsercvd RW 0x0	outtknepdis RW 0x0	setup RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doeptsiz10 0x450	Reserved	rxdpid RO 0x0		pktcnt RW 0x0										xfersize RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdma10 0x454	doepdma10 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	doepdma10 RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdmab10 0x45C	doepdmab10 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	doepdmab10 RO 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepctl11 0x460	epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	Reserved				stall RW 0x0	snp RW 0x0	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	usbactep RW 0x0	Reserved				mps RW 0x0										
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepint11 0x468	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	stuppktrcvd RW 0x0	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	outpkterr RW 0x0	Reserved	back2backsetup RW 0x0	stsphsercvd RW 0x0	outtknepdis RW 0x0	setup RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doeptsiz11 0x470	Reserved	rxdpid RO 0x0		pktcnt RW 0x0										xfersize RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdma11	doepdma11 RW 0x0															
0x474	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	doepdma11 RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdmab11	doepdmab11 RO 0x0															
0x47C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	doepdmab11 RO 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepctl12	epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	Reserved				stall RW 0x0	snp RW 0x0	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
0x480	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	usbactep RW 0x0	Reserved				mps RW 0x0										
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepint12	Reserved															
0x488	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	stuppktrcvd RW 0x0	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	outpkterr RW 0x0	Reserved	back2backsetup RW 0x0	stsphsercvd RW 0x0	outtknepdis RW 0x0	setup RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doeptsiz12	Reserved	rxdpid RO 0x0		pktcnt RW 0x0										xfersize RW 0x0		
0x490	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdma12 0x494	doepdma12 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	doepdma12 RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdmab12 0x49C	doepdmab12 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	doepdmab12 RO 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepctl13 0x4A0	epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	Reserved				stall RW 0x0	snp RW 0x0	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	usbactep RW 0x0	Reserved				mps RW 0x0										
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepint13 0x4A8	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	stuppktrcvd RW 0x0	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	outpkterr RW 0x0	Reserved	back2backsetup RW 0x0	stsphsercvd RW 0x0	outtknepdis RW 0x0	setup RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doeptsiz13 0x4B0	Reserved	rxdpid RO 0x0		pktcnt RW 0x0										xfersize RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdma13	doepdma13 RW 0x0															
0x4B4	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	doepdma13 RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdmab13	doepdmab13 RO 0x0															
0x4BC	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	doepdmab13 RO 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepctl14	epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	Reserved				stall RW 0x0	snp RW 0x0	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
0x4C0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	usbactep RW 0x0	Reserved				mps RW 0x0										
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
doepint14	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x4C8	stuppktrcvd RW 0x0	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	outpkterr RW 0x0	Reserved	back2backsetup RW 0x0	stsphsercvd RW 0x0	outtknepdis RW 0x0	setup RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doeptsiz14	Reserved	rxdpid RO 0x0		pktcnt RW 0x0										xfersize RW 0x0		
0x4D0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdma14 0x4D4	doepdma14 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	doepdma14 RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdmab14 0x4DC	doepdmab14 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	doepdmab14 RO 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepctl15 0x4E0	epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	Reserved				stall RW 0x0	snp RW 0x0	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	usbactep RW 0x0	Reserved				mps RW 0x0										
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepint15 0x4E8	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	stuppktrcvd RW 0x0	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	outpkterr RW 0x0	Reserved	back2backsetup RW 0x0	stsphsercvd RW 0x0	outtknepdis RW 0x0	setup RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doeptsiz15 0x4F0	Reserved	rxdpid RO 0x0		pktcnt RW 0x0										xfersize RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdma15	doepdma15 RW 0x0															
0x4F4	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	doepdma15 RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdmab15	doepdmab15 RO 0x0															
0x4FC	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	doepdmab15 RO 0x0															
i_usbotg_1_devgrp																
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dcfg	resvalid RW 0x2						perschintvl RW 0x0		descdma RW 0x0	Reserved						
0x0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	erraticintmsk RW 0x0	xcvrdly RW 0x0	endevoutnak RW 0x0	perfint RW 0x0		devaddr RW 0x0							ena32khzsusp RW 0x0	nzstsouthshk RW 0x0	devspd RW 0x0	
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dctl	Reserved														enctonbna RW 0x0	nakonbble RW 0x0
0x4	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ignrfrmnum RW 0x0	gmc RW 0x0		Reserved	pwronprgdone RW 0x0	cgoutnak WO 0x0	sgoutnak WO 0x0	cgnpinnak WO 0x0	sgnpinnak WO 0x0	tstctl RW 0x0			goutnaksts RO 0x0	gnpinnaksts RO 0x0	sftdiscon RW 0x1	rmtwkupsig RW 0x0

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved								devlnsts RO 0x0		soffn RO 0x0					
dsts 0x8	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	soffn RO 0x0								Reserved				errticerr RO 0x0	enumspd RO 0x1		suspsts RO 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
diepmsk 0x10	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		nakmsk RW 0x0	Reserved			bnainintrmsk RW 0x0	txfifoundrnmsk RW 0x0	Reserved	inepnakeffmsk RW 0x0	intknepmismsk RW 0x0	intkntxfempmsk RW 0x0	timeoutmsk RW 0x0	ahberrmsk RW 0x0	epdisbldmsk RW 0x0	xfercomplmsk RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
doepmsk 0x14	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	nyetmsk RW 0x0	nakmsk RW 0x0	bbleerrmsk RW 0x0	Reserved		bnaoutintrmsk RW 0x0	outpkterrmsk RW 0x0	Reserved	back2backsetup RW 0x0	stsphsercvdmsk RW 0x0	outtknepdismsk RW 0x0	setupmsk RW 0x0	ahberrmsk RW 0x0	epdisbldmsk RW 0x0	xfercomplmsk RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
daint 0x18	outepint15 RO 0x0	outepint14 RO 0x0	outepint13 RO 0x0	outepint12 RO 0x0	outepint11 RO 0x0	outepint10 RO 0x0	outepint9 RO 0x0	outepint8 RO 0x0	outepint7 RO 0x0	outepint6 RO 0x0	outepint5 RO 0x0	outepint4 RO 0x0	outepint3 RO 0x0	outepint2 RO 0x0	outepint1 RO 0x0	outepint0 RO 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	inepint15 RO 0x0	inepint14 RO 0x0	inepint13 RO 0x0	inepint12 RO 0x0	inepint11 RO 0x0	inepint10 RO 0x0	inepint9 RO 0x0	inepint8 RO 0x0	inepint7 RO 0x0	inepint6 RO 0x0	inepint5 RO 0x0	inepint4 RO 0x0	inepint3 RO 0x0	inepint2 RO 0x0	inepint1 RO 0x0	inepint0 RO 0x0

Register Address Offset	Bit Fields															
daintmsk 0x1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	outepmsk15 RW 0x0	outepmsk14 RW 0x0	outepmsk13 RW 0x0	outepmsk12 RW 0x0	outepmsk11 RW 0x0	outepmsk10 RW 0x0	outepmsk9 RW 0x0	outepmsk8 RW 0x0	outepmsk7 RW 0x0	outepmsk6 RW 0x0	outepmsk5 RW 0x0	outepmsk4 RW 0x0	outepmsk3 RW 0x0	outepmsk2 RW 0x0	outepmsk1 RW 0x0	outepmsk0 RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	inepmsk15 RW 0x0	inepmsk14 RW 0x0	inepmsk13 RW 0x0	inepmsk12 RW 0x0	inepmsk11 RW 0x0	inepmsk10 RW 0x0	inepmsk9 RW 0x0	inepmsk8 RW 0x0	inepmsk7 RW 0x0	inepmsk6 RW 0x0	inepmsk5 RW 0x0	inepmsk4 RW 0x0	inepmsk3 RW 0x0	inepmsk2 RW 0x0	inepmsk1 RW 0x0	inepmsk0 RW 0x0
dvbusdis 0x28	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	dvbusdis RW 0x17D7															
dvbuspulse 0x2C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				dvbuspulse RW 0x5B8											
dthrctl 0x30	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved				arbprken RW 0x1	Reserved	rxthrlen RW 0x8									rxthren RW 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved			ahbthrratio RW 0x0	txthrlen RW 0x8										isothren RW 0x0	nonisothren RW 0x0
diepempmsk 0x34	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ineptxfempmsk15 RW 0x0	ineptxfempmsk14 RW 0x0	ineptxfempmsk13 RW 0x0	ineptxfempmsk12 RW 0x0	ineptxfempmsk11 RW 0x0	ineptxfempmsk10 RW 0x0	ineptxfempmsk9 RW 0x0	ineptxfempmsk8 RW 0x0	ineptxfempmsk7 RW 0x0	ineptxfempmsk6 RW 0x0	ineptxfempmsk5 RW 0x0	ineptxfempmsk4 RW 0x0	ineptxfempmsk3 RW 0x0	ineptxfempmsk2 RW 0x0	ineptxfempmsk1 RW 0x0	ineptxfempmsk0 RW 0x0

Register Address Offset	Bit Fields															
diepctl0 0x100	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	epena RW 0x0	epdis RW 0x0	Reserved		snak WO 0x0	cnak WO 0x0	txfnum RW 0x0				stall RW 0x0	Reserved	eptype RO 0x0		naksts RO 0x0	Reserved
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	usbactep RO 0x1	Reserved													mps RW 0x0	
diepint0 0x108	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	nyetintrpt RW 0x0	nakintrpt RW 0x0	bleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	txfifoundrn RW 0x0	txfemp RO 0x1	inepnakeff RW 0x0	intknepmis RW 0x0	intkntxfemp RW 0x0	timeout RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0
dieptsiz0 0x110	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved											pktcnt RW 0x0		Reserved		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved									xfersize RW 0x0						
diepdma0 0x114	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	diepdma0 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	diepdma0 RW 0x0															
dtxfsts0 0x118	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ineptxfspcavail RO 0x2000															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepdmab0	diepdmab0 RO 0x0															
0x11C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	diepdmab0 RO 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepctl1	epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	txfnum RW 0x0				stall RW 0x0	Reserved	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
0x120	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	usbactep RW 0x0	Reserved				mps RW 0x0										
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
diepint1	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x128	Reserved	nyetintrpt RW 0x0	nakintrpt RW 0x0	bleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	txfifoundrn RW 0x0	txfemp RO 0x1	inepnakeff RW 0x0	intknepmis RW 0x0	intkntxfemp RW 0x0	timeout RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dieptsiz1	Reserved	mc RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
0x130	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepdma1	diepdma1 RW 0x0															
0x134	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	diepdma1 RW 0x0															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dtxfsts1 0x138	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ineptxfspcavail RO 0x2000															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepdmab1 0x13C	diepdmab1 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	diepdmab1 RO 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepctl2 0x140	epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	txfnum RW 0x0				stall RW 0x0	Reserved	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	usbactep RW 0x0	Reserved				mps RW 0x0										
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepint2 0x148	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	txfifoundrn RW 0x0	txfemp RO 0x1	inepnakeff RW 0x0	intknepmis RW 0x0	intkntxfemp RW 0x0	timeout RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dieptsiz2 0x150	Reserved	mc RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepdma2 0x154	diepdma2 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	diepdma2 RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dtxfsts2 0x158	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ineptxfspcavail RO 0x2000															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepdmab2 0x15C	diepdmab2 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	diepdmab2 RO 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepctl3 0x160	epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	txfnum RW 0x0				stall RW 0x0	Reserved	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	usbactep RW 0x0	Reserved				mps RW 0x0										
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepint3 0x168	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	txfifoundrn RW 0x0	txfemp RO 0x1	inepnakeff RW 0x0	intknepmis RW 0x0	intkntxfemp RW 0x0	timeout RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0

Register Address Offset	Bit Fields															
dieptsiz3 0x170	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	mc RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
diepdma3 0x174	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	diepdma3 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	diepdma3 RW 0x0															
dtxfsts3 0x178	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ineptxfspcavail RO 0x2000															
diepdmab3 0x17C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	diepdmab3 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	diepdmab3 RO 0x0															
diepctl4 0x180	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	txfnum RW 0x0				stall RW 0x0	Reserved	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	usbactep RW 0x0	Reserved				mps RW 0x0										

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
diepint4 0x188	Reserved	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	txfifoundrn RW 0x0	txfemp RO 0x1	inepnakeff RW 0x0	intknepmis RW 0x0	intkntxfemp RW 0x0	timeout RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	mc RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
dieptsiz4 0x190	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	diepdma4 RW 0x0															
diepdma4 0x194	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	diepdma4 RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
dtxfsts4 0x198	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ineptxfspcavail RO 0x2000															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	diepdmab4 RO 0x0															
diepdmab4 0x19C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	diepdmab4 RO 0x0															

Register Address Offset	Bit Fields															
diepctl5 0x1A0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	txfnum RW 0x0				stall RW 0x0	Reserved	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	usbactep RW 0x0	Reserved				mps RW 0x0										
diepint5 0x1A8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	nyetintrpt RW 0x0	nakintrpt RW 0x0	bleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	txfifoundrn RW 0x0	txfemp RO 0x1	inepnakeff RW 0x0	intknepmis RW 0x0	intkntxfemp RW 0x0	timeout RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0
dieptsiz5 0x1B0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	mc RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
diepdma5 0x1B4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	diepdma5 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	diepdma5 RW 0x0															
dtxfsts5 0x1B8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ineptxfspcavail RO 0x2000															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepdmab5	diepdmab5 RO 0x0															
0x1BC	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	diepdmab5 RO 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepctl6	epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	txfnum RW 0x0				stall RW 0x0	Reserved	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
0x1C0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	usbactep RW 0x0	Reserved				mps RW 0x0										
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
diepint6	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x1C8	Reserved	nyetintrpt RW 0x0	nakintrpt RW 0x0	bleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	txfifoundrn RW 0x0	txfemp RO 0x1	inepnakeff RW 0x0	intknepmis RW 0x0	intkntxfemp RW 0x0	timeout RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dieptsiz6	Reserved	mc RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
0x1D0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepdma6	diepdma6 RW 0x0															
0x1D4	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	diepdma6 RW 0x0															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dtxfsts6	Reserved															
0x1D8	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ineptxfspcavail RO 0x2000															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepdmab6	diepdmab6 RO 0x0															
0x1DC	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	diepdmab6 RO 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepctl7	epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	txfnum RW 0x0				stall RW 0x0	Reserved	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
0x1E0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	usbactep RW 0x0	Reserved				mps RW 0x0										
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepint7	Reserved															
0x1E8	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	txfifoundrn RW 0x0	txfemp RO 0x1	inepnakeff RW 0x0	intknepmis RW 0x0	intkntxfemp RW 0x0	timeout RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dieptsiz7	Reserved	mc RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
0x1F0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepdma7 0x1F4	diepdma7 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	diepdma7 RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dtxfsts7 0x1F8	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ineptxfspcavail RO 0x2000															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepdmab7 0x1FC	diepdmab7 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	diepdmab7 RO 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepctl8 0x200	epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	txfnum RW 0x0				stall RW 0x0	Reserved	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	usbactep RW 0x0	Reserved				mps RW 0x0										
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepint8 0x208	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	txfifoundrn RW 0x0	txfemp RO 0x1	inepnakeff RW 0x0	intknepmis RW 0x0	intkntxfemp RW 0x0	timeout RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dieptsiz8	Reserved	mc RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
0x210	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepdma8	diepdma8 RW 0x0															
0x214	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	diepdma8 RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dtxfsts8	Reserved															
0x218	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ineptxfspcavail RO 0x2000															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepdmab8	diepdmab8 RO 0x0															
0x21C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	diepdmab8 RO 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepctl9	epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	txfnum RW 0x0				stall RW 0x0	Reserved	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
0x220	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	usbactep RW 0x0	Reserved				mps RW 0x0										

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
diepint9 0x228	Reserved	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	txfifoundrn RW 0x0	txfemp RO 0x1	inepnakeff RW 0x0	intknepmis RW 0x0	intkntxfemp RW 0x0	timeout RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	mc RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
dieptsiz9 0x230	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	diepdma9 RW 0x0															
diepdma9 0x234	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	diepdma9 RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
dtxfsts9 0x238	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ineptxfspcavail RO 0x2000															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	diepdmab9 RO 0x0															
diepdmab9 0x23C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	diepdmab9 RO 0x0															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	txfnum RW 0x0				stall RW 0x0	Reserved	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
diepctl10 0x240	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	usbactep RW 0x0	Reserved				mps RW 0x0										
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
diepint10 0x248	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	nyetintrpt RW 0x0	nakintrpt RW 0x0	bleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	txfifoundrn RW 0x0	txfemp RO 0x1	inepnakeff RW 0x0	intknepmis RW 0x0	intkntxfemp RW 0x0	timeout RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	mc RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
dieptsiz10 0x250	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	diepdma10 RW 0x0															
diepdma10 0x254	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	diepdma10 RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
dtxfsts10 0x258	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ineptxfspcavail RO 0x2000															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepdmab10	diepdmab10 RO 0x0															
0x25C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	diepdmab10 RO 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepctl11	epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	txfnum RW 0x0				stall RW 0x0	Reserved	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
0x260	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	usbactep RW 0x0	Reserved				mps RW 0x0										
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
diepint11	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x268	Reserved	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	txfifoundrn RW 0x0	txfemp RO 0x1	inepnakeff RW 0x0	intknepmis RW 0x0	intkntxfemp RW 0x0	timeout RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dieptsiz11	Reserved	mc RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
0x270	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepdma11	diepdma11 RW 0x0															
0x274	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	diepdma11 RW 0x0															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dtxfsts11 0x278	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ineptxfspcavail RO 0x2000															
diepdmab11 0x27C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	diepdmab11 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	diepdmab11 RO 0x0															
diepctl12 0x280	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	txfnum RW 0x0				stall RW 0x0	Reserved	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	usbactep RW 0x0	Reserved				mps RW 0x0										
diepint12 0x288	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	txfifoundrn RW 0x0	txfemp RO 0x1	inepnakeff RW 0x0	intknepmis RW 0x0	intkntxfemp RW 0x0	timeout RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0
dieptsiz12 0x290	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	mc RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepdma12 0x294	diepdma12 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	diepdma12 RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dtxfsts12 0x298	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ineptxfspcavail RO 0x2000															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepdmab12 0x29C	diepdmab12 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	diepdmab12 RO 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepctl13 0x2A0	epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	txfnum RW 0x0				stall RW 0x0	Reserved	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	usbactep RW 0x0	Reserved				mps RW 0x0										
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepint13 0x2A8	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	txfifoundrn RW 0x0	txfemp RO 0x1	inepnakeff RW 0x0	intknepmis RW 0x0	intkntxfemp RW 0x0	timeout RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dieptsiz13	Reserved	mc RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
0x2B0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepdma13	diepdma13 RW 0x0															
0x2B4	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	diepdma13 RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dtxfsts13	Reserved															
0x2B8	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ineptxfspcavail RO 0x2000															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepdmab13	diepdmab13 RO 0x0															
0x2BC	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	diepdmab13 RO 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepctl14	epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	txfnum RW 0x0				stall RW 0x0	Reserved	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
0x2C0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	usbactep RW 0x0	Reserved				mps RW 0x0										

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
diepint14 0x2C8	Reserved	nyetintrpt RW 0x0	nakintrpt RW 0x0	bleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	txfifoundrn RW 0x0	txfemp RO 0x1	inepnakeff RW 0x0	intknepmis RW 0x0	intkntxfemp RW 0x0	timeout RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	mc RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
dieptsiz14 0x2D0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	diepdma14 RW 0x0															
diepdma14 0x2D4	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	diepdma14 RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
dtxfsts14 0x2D8	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ineptxfspcavail RO 0x2000															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	diepdmab14 RO 0x0															
diepdmab14 0x2DC	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	diepdmab14 RO 0x0															

Register Address Offset	Bit Fields															
diepctl15 0x2E0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	txfnum RW 0x0				stall RW 0x0	Reserved	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	usbactep RW 0x0	Reserved				mps RW 0x0										
diepint15 0x2E8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	nyetintrpt RW 0x0	nakintrpt RW 0x0	bleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	txfifoundrn RW 0x0	txfemp RO 0x1	inepnakeff RW 0x0	intknepmis RW 0x0	intkntxfemp RW 0x0	timeout RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0
dieptsiz15 0x2F0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved	mc RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
diepdma15 0x2F4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	diepdma15 RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	diepdma15 RW 0x0															
dtxfsts15 0x2F8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ineptxfspcavail RO 0x2000															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepdmab15	diepdmab15 RO 0x0															
0x2FC	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	diepdmab15 RO 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepctl0	epena RW 0x0	epdis RO 0x0	Reserved		snak WO 0x0	cnak WO 0x0	Reserved				stall RW 0x0	snp RW 0x0	eptype RO 0x0		naksts RO 0x0	Reserved
0x300	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	usbactep RO 0x1	Reserved													mps RO 0x0	
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepint0	Reserved															
0x308	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	stuppktrcvd RW 0x0	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	outpkterr RW 0x0	Reserved	back2backsetup RW 0x0	stsphsercvd RW 0x0	outtknepdis RW 0x0	setup RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doeptsiz0	Reserved	supcnt RW 0x0		Reserved									pktcnt RW 0x0	Reserved		
0x310	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved									xfersize RW 0x0						
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdma0	doepdma0 RW 0x0															
0x314	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	doepdma0 RW 0x0															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdmab0	doepdmab0 RO 0x0															
0x31C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	doepdmab0 RO 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepctl1	epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	Reserved				stall RW 0x0	snp RW 0x0	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
0x320	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	usbactep RW 0x0	Reserved				mps RW 0x0										
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepint1	Reserved															
0x328	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	stuppktrcvd RW 0x0	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	outpkterr RW 0x0	Reserved	back2backsetup RW 0x0	stsphsercvd RW 0x0	outtknepdis RW 0x0	setup RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doeptsiz1	Reserved	rxdpid RO 0x0		pktcnt RW 0x0										xfersize RW 0x0		
0x330	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdma1	doepdma1 RW 0x0															
0x334	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	doepdma1 RW 0x0															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdmab1	doepdmab1 RO 0x0															
0x33C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	doepdmab1 RO 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepctl2	epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	Reserved				stall RW 0x0	snp RW 0x0	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
0x340	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	usbactep RW 0x0	Reserved				mps RW 0x0										
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
doepint2	stuppktrcvd RW 0x0	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	outpkterr RW 0x0	Reserved	back2backsetup RW 0x0	stsphsercvd RW 0x0	outtknepdis RW 0x0	setup RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0
0x348																
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doeptsiz2	Reserved	rxdpid RO 0x0		pktcnt RW 0x0										xfersize RW 0x0		
0x350	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdma2	doepdma2 RW 0x0															
0x354	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	doepdma2 RW 0x0															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdmab2	doepdmab2 RO 0x0															
0x35C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	doepdmab2 RO 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepctl3	epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	Reserved				stall RW 0x0	snp RW 0x0	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
0x360	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	usbactep RW 0x0	Reserved				mps RW 0x0										
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepint3	Reserved															
0x368	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	stuppktrcvd RW 0x0	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	outpkterr RW 0x0	Reserved	back2backsetup RW 0x0	stsphsercvd RW 0x0	outtknepdis RW 0x0	setup RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doeptsiz3	Reserved	rxdpid RO 0x0		pktcnt RW 0x0										xfersize RW 0x0		
0x370	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdma3	doepdma3 RW 0x0															
0x374	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	doepdma3 RW 0x0															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdmab3	doepdmab3 RO 0x0															
0x37C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	doepdmab3 RO 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepctl4	epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	Reserved				stall RW 0x0	snp RW 0x0	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
0x380	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	usbactep RW 0x0	Reserved				mps RW 0x0										
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
doepint4	stuppktrcvd RW 0x0	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	outpkterr RW 0x0	Reserved	back2backsetup RW 0x0	stsphsercvd RW 0x0	outtknepdis RW 0x0	setup RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0
0x388																
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doeptsiz4	Reserved	rxdpid RO 0x0		pktcnt RW 0x0										xfersize RW 0x0		
0x390	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdma4	doepdma4 RW 0x0															
0x394	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	doepdma4 RW 0x0															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdmab4	doepdmab4 RO 0x0															
0x39C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	doepdmab4 RO 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepctl5	epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	Reserved				stall RW 0x0	snp RW 0x0	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
0x3A0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	usbactep RW 0x0	Reserved				mps RW 0x0										
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepint5	Reserved															
0x3A8	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	stuppktrcvd RW 0x0	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	outpkterr RW 0x0	Reserved	back2backsetup RW 0x0	stsphsercvd RW 0x0	outtknepdis RW 0x0	setup RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doeptsiz5	Reserved	rxdpid RO 0x0		pktcnt RW 0x0										xfersize RW 0x0		
0x3B0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdma5	doepdma5 RW 0x0															
0x3B4	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	doepdma5 RW 0x0															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdmab5	doepdmab5 RO 0x0															
0x3BC	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	doepdmab5 RO 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepctl6	epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	Reserved				stall RW 0x0	snp RW 0x0	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
0x3C0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	usbactep RW 0x0	Reserved				mps RW 0x0										
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
doepint6	stuppktrcvd RW 0x0	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	outpkterr RW 0x0	Reserved	back2backsetup RW 0x0	stsphsercvd RW 0x0	outtknepdis RW 0x0	setup RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0
0x3C8																
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doeptsiz6	Reserved	rxdpid RO 0x0		pktcnt RW 0x0										xfersize RW 0x0		
0x3D0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdma6	doepdma6 RW 0x0															
0x3D4	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	doepdma6 RW 0x0															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdmab6	doepdmab6 RO 0x0															
0x3DC	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	doepdmab6 RO 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepctl7	epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	Reserved				stall RW 0x0	snp RW 0x0	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
0x3E0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	usbactep RW 0x0	Reserved				mps RW 0x0										
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepint7	Reserved															
0x3E8	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	stuppktrcvd RW 0x0	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	outpkterr RW 0x0	Reserved	back2backsetup RW 0x0	stsphsercvd RW 0x0	outtknepdis RW 0x0	setup RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doeptsiz7	Reserved	rxdpid RO 0x0		pktcnt RW 0x0										xfersize RW 0x0		
0x3F0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdma7	doepdma7 RW 0x0															
0x3F4	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	doepdma7 RW 0x0															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdmab7	doepdmab7 RO 0x0															
0x3FC	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	doepdmab7 RO 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepctl8	epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	Reserved				stall RW 0x0	snp RW 0x0	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
0x400	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	usbactep RW 0x0	Reserved				mps RW 0x0										
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
doepint8	stuppktrcvd RW 0x0	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	outpkterr RW 0x0	Reserved	back2backsetup RW 0x0	stsphsercvd RW 0x0	outtknepdis RW 0x0	setup RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0
0x408																
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doeptsiz8	Reserved	rxdpid RO 0x0		pktcnt RW 0x0										xfersize RW 0x0		
0x410	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdma8	doepdma8 RW 0x0															
0x414	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	doepdma8 RW 0x0															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdmab8	doepdmab8 RO 0x0															
0x41C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	doepdmab8 RO 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepctl9	epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	Reserved				stall RW 0x0	snp RW 0x0	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
0x420	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	usbactep RW 0x0	Reserved				mps RW 0x0										
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepint9	Reserved															
0x428	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	stuppktrcvd RW 0x0	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	outpkterr RW 0x0	Reserved	back2backsetup RW 0x0	stsphsercvd RW 0x0	outtknepdis RW 0x0	setup RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doeptsiz9	Reserved	rxdpid RO 0x0		pktcnt RW 0x0										xfersize RW 0x0		
0x430	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdma9	doepdma9 RW 0x0															
0x434	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	doepdma9 RW 0x0															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdmab9	doepdmab9 RO 0x0															
0x43C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	doepdmab9 RO 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepctl10	epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	Reserved				stall RW 0x0	snp RW 0x0	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
0x440	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	usbactep RW 0x0	Reserved				mps RW 0x0										
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
doepint10 0x448	stuppktrcvd RW 0x0	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	outpkterr RW 0x0	Reserved	back2backsetup RW 0x0	stsphsercvd RW 0x0	outtknepdis RW 0x0	setup RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doeptsiz10	Reserved	rxdpid RO 0x0		pktcnt RW 0x0										xfersize RW 0x0		
0x450	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdma10	doepdma10 RW 0x0															
0x454	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	doepdma10 RW 0x0															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdmab10	doepdmab10 RO 0x0															
0x45C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	doepdmab10 RO 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepctl11	epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	Reserved				stall RW 0x0	snp RW 0x0	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
0x460	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	usbactep RW 0x0	Reserved				mps RW 0x0										
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepint11	Reserved															
0x468	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	stuppktrcvd RW 0x0	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	outpkterr RW 0x0	Reserved	back2backsetup RW 0x0	stsphsercvd RW 0x0	outtknepdis RW 0x0	setup RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doeptsiz11	Reserved	rxdpid RO 0x0		pktcnt RW 0x0										xfersize RW 0x0		
0x470	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdma11	doepdma11 RW 0x0															
0x474	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	doepdma11 RW 0x0															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdmab11	doepdmab11 RO 0x0															
0x47C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	doepdmab11 RO 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepctl12	epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	Reserved				stall RW 0x0	snp RW 0x0	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
0x480	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	usbactep RW 0x0	Reserved				mps RW 0x0										
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepint12	Reserved															
0x488	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	stuppktrcvd RW 0x0	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	outpkterr RW 0x0	Reserved	back2backsetup RW 0x0	stsphsercvd RW 0x0	outtknepdis RW 0x0	setup RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doeptsiz12	Reserved	rxdpid RO 0x0		pktcnt RW 0x0										xfersize RW 0x0		
0x490	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdma12	doepdma12 RW 0x0															
0x494	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	doepdma12 RW 0x0															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdmab12	doepdmab12 RO 0x0															
0x49C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	doepdmab12 RO 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepctl13	epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	Reserved				stall RW 0x0	snp RW 0x0	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
0x4A0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	usbactep RW 0x0	Reserved				mps RW 0x0										
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepint13	Reserved															
0x4A8	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	stuppktrcvd RW 0x0	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	outpkterr RW 0x0	Reserved	back2backsetup RW 0x0	stsphsercvd RW 0x0	outtknepdis RW 0x0	setup RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doeptsiz13	Reserved	rxdpid RO 0x0		pktcnt RW 0x0										xfersize RW 0x0		
0x4B0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdma13	doepdma13 RW 0x0															
0x4B4	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	doepdma13 RW 0x0															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdmab13	doepdmab13 RO 0x0															
0x4BC	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	doepdmab13 RO 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepctl14	epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	Reserved				stall RW 0x0	snp RW 0x0	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
0x4C0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	usbactep RW 0x0	Reserved				mps RW 0x0										
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
doepint14	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x4C8	stuppktrcvd RW 0x0	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	outpkterr RW 0x0	Reserved	back2backsetup RW 0x0	stsphsercvd RW 0x0	outtknepdis RW 0x0	setup RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doeptsiz14	Reserved	rxdpid RO 0x0		pktcnt RW 0x0										xfersize RW 0x0		
0x4D0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdma14	doepdma14 RW 0x0															
0x4D4	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	doepdma14 RW 0x0															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdmab14	doepdmab14 RO 0x0															
0x4DC	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	doepdmab14 RO 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepctl15	epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	Reserved				stall RW 0x0	snp RW 0x0	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
0x4E0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	usbactep RW 0x0	Reserved				mps RW 0x0										
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
doepint15	stuppktrcvd RW 0x0	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	outpkterr RW 0x0	Reserved	back2backsetup RW 0x0	stsphsercvd RW 0x0	outtknepdis RW 0x0	setup RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0
0x4E8																
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doeptsiz15	Reserved	rxdpid RO 0x0		pktcnt RW 0x0										xfersize RW 0x0		
0x4F0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	xfersize RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdma15	doepdma15 RW 0x0															
0x4F4	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	doepdma15 RW 0x0															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdmab15	doepdmab15 RO 0x0															
0x4FC	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	doepdmab15 RO 0x0															

dcfg

Device Configuration Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00800
i_usbotg_1_devgrp	0xFFB40800	0xFFB40800

Offset: 0x0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
resvalid RW 0x2						perschintvl RW 0x0		descdma RW 0x0	Reserved						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
erraticintmsk RW 0x0	xcvrdly RW 0x0	endevoutnak RW 0x0	perfrint RW 0x0			devaddr RW 0x0						ena32khzsusp RW 0x0	nzstsouthshk RW 0x0	devspd RW 0x0	

dcfg Fields

Bit	Name	Description	Access	Reset
31:26	resvalid	Resume Validation Period (ResValid) This field is effective only when DCFG.Ena32KHzSusp is set. It will control the resume period when the core resumes from suspend. The core counts for “ResValid” number of clock cycles to detect a valid resume when this is set	RW	0x2

Bit	Name	Description	Access	Reset
25:24	perschintvl	Periodic Scheduling Interval (PerSchIntvl) PerSchIntvl must be programmed only For Scatter/Gather DMA mode. Description: This field specifies the amount of time the Internal DMA engine must allocate For fetching periodic IN endpoint data. Based on the number of periodic endpoints, this value must be specified as 25,50 or 75% of (micro) frame. When any periodic endpoints are active, the internal DMA engine allocates the specified amount of time in fetching periodic IN endpoint data . When no periodic endpoints are active, Then the internal DMA engine services non-periodic endpoints, ignoring this field. After the specified time within a (micro)frame, the DMA switches to fetching For non-periodic endpoints. 2'b00: 25% of (micro)frame. 2'b01: 50% of (micro)frame. 2'b10: 75% of (micro)frame. 2'b11: Reserved. Reset: 2'b00 **Value** **Description** 0x0 MF25 0x1 MF50 0x2 MF75 0x3 RESERVED	RW	0x0

Bit	Name	Description	Access	Reset
23	descdma	Enable Scatter/gather DMA in device mode (DescDMA). When the Scatter/Gather DMA option selected during configuration of the RTL, the application can Set this bit during initialization to enable the Scatter/Gather DMA operation. NOTE: This bit must be modified only once after a reset. The following combinations are available For programming: GAHBCFG.DMAEn=0,DCFG.DescDMA=0 => Slave mode GAHBCFG.DMAEn=0,DCFG.DescDMA=1 => Invalid GAHBCFG.DMAEn=1,DCFG.DescDMA=0 => Buffered DMA mode GAHBCFG.DMAEn=1,DCFG.DescDMA=1 => Scatter/Gather DMA mode **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
15	erraticintmsk	Erratic Error Interrupt Mask 1'b1: Mask early suspend interrupt on erratic error 1'b0: Early suspend interrupt is generated on erratic error	RW	0x0
14	xcvrdly	1'b1: Enable delay between xcvr_sel and txvalid during Device chirp 1'b0: No delay between xcvr_sel and txvalid during Device chirp	RW	0x0

Bit	Name	Description	Access	Reset
13	endevoutnak	Enable Device OUT NAK (EnDevOutNak) This bit enables setting NAK for Bulk OUT endpoints after the transfer is completed for Device mode Descriptor DMA 1'b0 : The core does not set NAK after Bulk OUT transfer complete 1'b1 : The core sets NAK after Bulk OUT transfer complete It is one time programmable after reset like any other DCFG register bits. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
12:11	perfrint	Periodic Frame Interval (PerFrInt) Indicates the time within a (micro) frame at which the application must be notified using the End Of Periodic Frame Interrupt. This can be used to determine If all the isochronous traffic For that (micro)frame is complete. 2'b00: 80% of the (micro)frame interval 2'b01: 85% 2'b10: 90% 2'b11: 95% **Value** **Description** 0x0 EOPF80 0x1 EOPF85 0x2 EOPF90 0x3 EOPF95	RW	0x0
10:4	devaddr	Device Address (DevAddr) The application must program this field after every SetAddress control command.	RW	0x0

Bit	Name	Description	Access	Reset
3	ena32khzsusp	Enable 32 KHz Suspend mode (Ena32KHzSusp) This bit can be set only if FS PHY interface is selected. Else, this bit needs to be set to zero. When FS PHY interface is chosen and this bit is set, the core expects that the PHY clock during Suspend is switched from 48 MHz to 32 KHz. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
2	nzstsouthshk	Non-Zero-Length Status OUT Handshake (NZStsOUTHShk) The application can use this field to select the handshake the core sends on receiving a nonzero-length data packet during the OUT transaction of a control transfer's Status stage. 1'b1: Send a STALL handshake on a nonzero-length status OUT transaction and do not send the received OUT packet to the application. 1'b0: Send the received OUT packet to the application (zerolength or nonzero-length) and send a handshake based on the NAK and STALL bits For the endpoint in the Device Endpoint Control register. **Value** **Description** 0x0 SENDOUT 0x1 SENDSTALL	RW	0x0

Bit	Name	Description	Access	Reset
1:0	devspd	Device Speed (DevSpd) Indicates the speed at which the application requires the core to enumerate, or the maximum speed the application can support. However, the actual bus speed is determined only after the chirp sequence is completed, and is based on the speed of the USB host to which the core is connected. See “Device Initialization” . 2'b00: High speed (USB 2.0 PHY clock is 30 MHz or 60 MHz) 2'b01: Full speed (USB 2.0 PHY clock is 30 MHz or 60 MHz) 2'b10: Low speed (USB 1.1 transceiver clock is 6 MHz). If you select 6 MHz LS mode, you must do a soft reset. 2'b11: Full speed (USB 1.1 transceiver clock is 48 MHz) **Value** — **Description** 0x0 — USBHS20 0x1 — USBFS20 0x2 — USBLS116 0x3 — USBLS1148	RW	0x0

dctl

Device Control Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00804
i_usbotg_1_devgrp	0xFFB40800	0xFFB40804

Offset: 0x4

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved														encontonbna RW 0x0	nakonbble RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ignrfrmnum RW 0x0	gmc RW 0x0		Reserved	pwronprgdone RW 0x0	cgoutnak WO 0x0	sgoutnak WO 0x0	cgnpinnak WO 0x0	sgnpinnak WO 0x0	tstctl RW 0x0			goutnaksts RO 0x0	gnpinnaksts RO 0x0	sftdiscon RW 0x1	rmtwkupsig RW 0x0

dctl Fields

Bit	Name	Description	Access	Reset
17	encontonbna	Enable Continue on BNA (EnContOnBNA) This bit enables the DWC_otg core to continue on BNA for Bulk OUT endpoints. With this feature enabled, when a Bulk OUT endpoint receives a BNA interrupt the core starts processing the descriptor that caused the BNA interrupt after the endpoint re-enables the endpoint. 1'b0: After receiving BNA interrupt,the core disables the endpoint. When the endpoint is re-enabled by the application,the core starts processing from the DOEPDMA descriptor. 1'b1: After receiving BNA interrupt, the core disables the endpoint. When the endpoint is re-enabled by the application, the core starts processing from the descriptor that received the BNA interrupt. This bit is valid only when OTG_EN_DESC_DMA == 1'b1. It is a one-time programmable after reset bit like any other DCTL register bits.	RW	0x0
16	nakonbble	NAK on Babble Error (NakOnBble) Set NAK automatically on babble (NakOnBble). The core sets NAK automatically for the endpoint on which babble is received. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
15	ignrfrmnum	Ignore Frame number For Isochronous End points (IgnrFrmNum) Do NOT program IgnrFrmNum bit to 1'b1 when the core is operating in threshold mode. Note: When Scatter/Gather DMA mode is enabled this feature is not applicable to High Speed, High bandwidth transfers. When this bit is enabled, there must be only one packet per descriptor. 0: The core transmits the packets only in the frame number in which they are intended to be transmitted. 1: The core ignores the frame number, sending packets immediately as the packets are ready. In Scatter/Gather DMA mode, if this bit is enabled, the packets are not flushed when a ISOC IN token is received for an elapsed frame. When Scatter/Gather DMA mode is disabled, this field is used by the application to enable periodic transfer interrupt. The application can program periodic endpoint transfers for multiple (micro) frames. 0: periodic transfer interrupt feature is disabled, application needs to program transfers for periodic endpoints every (micro) frame 1: periodic transfer interrupt feature is enabled, application can program transfers for multiple (micro)frames for periodic endpoints. In non Scatter/Gather DMA mode the application will receive transfer complete interrupt after transfers for multiple (micro)frames are completed. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
14:13	gmc	Global Multi Count (GMC) GMC must be programmed only once after initialization. Applicable only For Scatter/Gather DMA mode. This indicates the number of packets to be serviced For that end point before moving to the next end point. It is only For non-periodic end points. 2'b00: Invalid. 2'b01: 1 packet. 2'b10: 2 packets. 2'b11: 3 packets. When Scatter/Gather DMA mode is disabled, this field is reserved. and reads 2'b00. **Value** **Description** 0x0 NOTVALID 0x1 ONEPACKET 0x2 TWOPACKET 0x3 THREEPACKET	RW	0x0
11	pwronprgdone	Power-On Programming Done (PWROnPrg-Done) The application uses this bit to indicate that register programming is completed after a wake-up from Power Down mode. **Value** **Description** 0x0 NOTDONE 0x1 DONE	RW	0x0
10	cgoutnak	Clear Global OUT NAK (CGOUTNak) A write to this field clears the Global OUT NAK. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	WO	0x0

Bit	Name	Description	Access	Reset
9	sgoutnak	Set Global OUT NAK (SGOUTNak) A write to this field sets the Global OUT NAK. The application uses this bit to send a NAK handshake on all OUT endpoints. The application must Set the this bit only after making sure that the Global OUT NAK Effective bit in the Core Interrupt Register (GINTSTS.GOUTNakEff) is cleared. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	WO	0x0
8	cgnpinnak	Clear Global Non-periodic IN NAK (CGNPInNak) A write to this field clears the Global Non-periodic IN NAK. **Value** **Description** 0x0 DISABLE 0x1 ENABLE	WO	0x0

Bit	Name	Description	Access	Reset
7	sgnpinnak	Set Global Non-periodic IN NAK (SGNPInNak) A write to this field sets the Global Non-periodic IN NAK.The application uses this bit to send a NAK handshake on all nonperiodic IN endpoints. The core can also Set this bit when a timeout condition is detected on a non-periodic endpoint in shared FIFO operation. The application must Set this bit only after making sure that the Global IN NAK Effective bit in the Core Interrupt Register (GINTSTS.GINNakEff) is cleared **Value** **Description** 0x0 DISABLED 0x1 ENABLED	WO	0x0
6:4	tstctl	Test Control (TstCtl) 3'b000: Test mode disabled 3'b001: Test_J mode 3'b010: Test_K mode 3'b011: Test_SE0_NAK mode 3'b100: Test_Packet mode 3'b101: Test_Force_Enable Others: Reserved **Value** **Description** 0x0 DISABLED 0x1 TESTJ 0x2 TESTK 0x3 TESTSN 0x4 TESTPM 0x5 TESTFE	RW	0x0

Bit	Name	Description	Access	Reset
3	goutnaksts	Global OUT NAK Status (GOUTNakSts) 1'b0: A handshake is sent based on the FIFO Status and the NAK and STALL bit settings. 1'b1: No data is written to the RxFIFO, irrespective of space availability. Sends a NAK handshake on all packets, except on SETUP transactions. All isochronous OUT packets are dropped. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0
2	gnpinnaksts	Global Non-periodic IN NAK Status (GNPINNakSts) 1'b0: A handshake is sent out based on the data availability in the transmit FIFO. 1'b1: A NAK handshake is sent out on all non-periodic IN endpoints, irrespective of the data availability in the transmit FIFO. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0

Bit	Name	Description	Access	Reset
1	sftdiscon	Soft Disconnect (SftDiscon) The application uses this bit to signal the DWC_otg core to do a soft disconnect. As long as this bit is Set, the host does not see that the device is connected, and the device does not receive signals on the USB. The core stays in the disconnected state until the application clears this bit. The minimum duration For which the core must keep this bit Set is specified in Table 5-46. 1'b0: Normal operation. When this bit is cleared after a soft disconnect, the core drives the phy_opmode_o signal on the UTMI+ to 2'b00, which generates a device connect event to the USB host. When the device is reconnected, the USB host restarts device enumeration. 1'b1: The core drives the phy_opmode_o signal on the UTMI+ to 2'b01, which generates a device disconnect event to the USB host. Note: This bit can be also used for ULPI/FS Serial interfaces. Note: This bit is not impacted by a soft reset. **Value** **Description** 0x0 NODISCONNECT 0x1 DISCONNECT	RW	0x1

Bit	Name	Description	Access	Reset
0	rmtwkupsig	Remote Wakeup Signaling (RmtWkUpSig) When the application sets this bit, the core initiates remote signaling to wake up the USB host. The application must Set this bit to instruct the core to exit the Suspend state. As specified in the USB 2.0 specification, the application must clear this bit 1-15 ms after setting it. Remote Wakeup Signaling (RmtWkUpSig) When LPM is enabled, In L1 state the behavior of this bit is as follows: When the application sets this bit, the core initiates L1 remote signaling to wake up the USB host. The application must set this bit to instruct the core to exit the Sleep state. As specified in the LPM specification, the hardware will automatically clear this bit after a time of 50 micro sec (TL1DevDrvResume) after set by application. Application should not set this bit when GLPMCFG bRemoteWake from the previous LPM transaction was zero. **Value** **Description** 0x0 NOEXIT 0x1 EXIT	RW	0x0

dsts

Device Status Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00808
i_usbotg_1_devgrp	0xFFB40800	0xFFB40808

Offset: 0x8

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								devlnsts RO 0x0		soffn RO 0x0					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
soffn RO 0x0								Reserved				errticerr RO 0x0	enumspd RO 0x1		suspsts RO 0x0

dsts Fields

Bit	Name	Description	Access	Reset
23:22	devlnsts	Device Line Status (DevLnSts) Indicates the current logic level USB data lines DevLnSts[1]: Logic level of D+ DevLnSts[0]: Logic level of D-	RO	0x0
21:8	soffn	Frame or Microframe Number of the Received SOF (SOFFN) When the core is operating at high speed, this field contains a microframe number. When the core is operating at full or low speed, this field contains a Frame number. Note: This register may return a non zero value if read immediately after power on reset. In case the register bit reads non zero immediately after power on reset it does not indicate that SOF has been received from the host. The read value of this interrupt is valid only after a valid connection between host and device is established.	RO	0x0

Bit	Name	Description	Access	Reset
3	errticerr	Erratic Error (ErrticErr) The core sets this bit to report any erratic errors (phy_rxvalid_i/phy_rxvldh_i or phy_rxactive_i is asserted For at least 2 ms, due to PHY error) seen on the UTMI+ . Due to erratic errors, the DWC_otg core goes into Suspended state and an interrupt is generated to the application with Early Suspend bit of the Core Interrupt register (GINTSTS.ErlySusp). If the early suspend is asserted due to an erratic error, the application can only perform a soft disconnect recover. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0
2:1	enumspd	Enumerated Speed (EnumSpd) Indicates the speed at which the DWC_otg core has come up after speed detection through a chirp sequence. 2'b00: High speed (PHY clock is running at 30 or 60 MHz) 2'b01: Full speed (PHY clock is running at 30 or 60 MHz) 2'b10: Low speed (PHY clock is running at 6 MHz) 2'b11: Full speed (PHY clock is running at 48 MHz) Low speed is not supported For devices using a UTMI+ PHY. **Value** **Description** 0x0 HS3060 0x1 FS3060 0x2 LS6 0x3 FS48	RO	0x1

Bit	Name	Description	Access	Reset
0	suspsts	Suspend Status (SuspSts) In Device mode, this bit is Set as long as a Suspend condition is detected on the USB. The core enters the Suspended state when there is no activity on the phy_line_state_i signal For an extended period of time. The core comes out of the suspend: When there is any activity on the phy_line_state_i signal When the application writes to the Remote Wakeup Signaling bit in the Device Control register (DCTL.RmtWkUpSig). **Value** / **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RO	0x0

diepmsk

Device IN Endpoint Common Interrupt Mask Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00810
i_usbotg_1_devgrp	0xFFB40800	0xFFB40810

Offset: 0x10

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		nakmsk RW 0x0	Reserved			bnainintrmsk RW 0x0	txfifoundrnmsk RW 0x0	Reserved	inepnakeffmsk RW 0x0	intknepmismsk RW 0x0	intkntxfempmsk RW 0x0	timeoutmsk RW 0x0	ahberrmsk RW 0x0	epdisbldmsk RW 0x0	xfercomplmsk RW 0x0

diepmsk Fields

Bit	Name	Description	Access	Reset
13	nakmsk	NAK interrupt Mask (NAKMsk) Value: Description 0x0: MASK 0x1: NOMASK	RW	0x0
9	bnainintrmsk	BNA interrupt Mask (BNAInIntrMsk) Value: Description 0x0: MASK 0x1: NOMASK	RW	0x0
8	txfifoundrnmsk	Fifo Underrun Mask (TxfifoUndrnMsk) Value: Description 0x0: MASK 0x1: NOMASK	RW	0x0
6	inepnakeffmsk	IN Endpoint NAK Effective Mask (INEPNakEffMsk) Value: Description 0x0: MASK 0x1: NOMASK	RW	0x0

Bit	Name	Description	Access	Reset
5	intknepmismsk	IN Token received with EP Mismatch Mask (INTknEPMisMsk) Value: 0x0 = MASK; 0x1 = NOMASK	RW	0x0
4	intkntxfempmsk	IN Token Received When TxFIFO Empty Mask (INTknTXFEmpMsk) Value: 0x0 = MASK; 0x1 = NOMASK	RW	0x0
3	timeoutmsk	Timeout Condition Mask (TimeOUTMsk) (Non-isochronous endpoints) Value: 0x0 = MASK; 0x1 = NOMASK	RW	0x0
2	ahberrmsk	AHB Error Mask (AHBErrMsk) Value: 0x0 = MASK; 0x1 = NOMASK	RW	0x0
1	epdisbldmsk	Endpoint Disabled Interrupt Mask (EPDisbldMsk) Value: 0x0 = MASK; 0x1 = NOMASK	RW	0x0

Bit	Name	Description	Access	Reset
0	xfercomplmsk	Transfer Completed Interrupt Mask (XferComplMsk) Value: Description 0x0: MASK 0x1: NOMASK	RW	0x0

doepmsk

Device OUT Endpoint Common Interrupt Mask Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00814
i_usbotg_1_devgrp	0xFFB40800	0xFFB40814

Offset: 0x14

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	nyetmsk RW 0x0	nakmsk RW 0x0	bbleerrmsk RW 0x0	Reserved		bnaoutintrmsk RW 0x0	outpkterrmsk RW 0x0	Reserved	back2backsetup RW 0x0	stsphsercvdmsk RW 0x0	outtknepdismsk RW 0x0	setupmsk RW 0x0	ahberrmsk RW 0x0	epdisbldmsk RW 0x0	xfercomplmsk RW 0x0

doepmsk Fields

Bit	Name	Description	Access	Reset
14	nyetmsk	NYET interrupt Mask (NYETMsk) Value: Description 0x0: MASK 0x1: NOMASK	RW	0x0
13	nakmsk	NAK interrupt Mask (NAKMsk) Value: Description 0x0: MASK 0x1: NOMASK	RW	0x0
12	bbleerrmsk	Babble Error interrupt Mask (BbleErrMsk) Value: Description 0x0: MASK 0x1: NOMASK	RW	0x0
9	bnaoutintrmsk	BNA interrupt Mask (BnaOutIntrMsk) Value: Description 0x0: MASK 0x1: NOMASK	RW	0x0
8	outpkterrmsk	OUT Packet Error Mask (OutPktErrMsk) Value: Description 0x0: MASK 0x1: NOMASK	RW	0x0

Bit	Name	Description	Access	Reset
6	back2backsetup	Back-to-Back SETUP Packets Received Mask (Back2BackSETup) Applies to control OUT endpoints only. Value: 0x0 MASK; 0x1 NOMASK	RW	0x0
5	stsphsercvdmsk	Status Phase Received Mask (StsPhseRcvdMsk) Applies to control OUT endpoints only.	RW	0x0
4	outtknepdismsk	OUT Token Received when Endpoint Disabled Mask (OUTTknEPdisMsk) Applies to control OUT endpoints only. Value: 0x0 MASK; 0x1 NOMASK	RW	0x0
3	setupmsk	SETUP Phase Done Mask (SetUPMsk) Applies to control endpoints only. Value: 0x0 MASK; 0x1 NOMASK	RW	0x0
2	ahberrmsk	AHB Error (AHBErrMsk) Value: 0x0 MASK; 0x1 NOMASK	RW	0x0

Bit	Name	Description	Access	Reset
1	epdisbldmsk	Endpoint Disabled Interrupt Mask (EPDisbldMsk) Value: Description 0x0: MASK 0x1: NOMASK	RW	0x0
0	xfercomplmsk	Transfer Completed Interrupt Mask (XferComplMsk) Value: Description 0x0: MASK 0x1: NOMASK	RW	0x0

daint

Device All Endpoints Interrupt Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00818
i_usbotg_1_devgrp	0xFFB40800	0xFFB40818

Offset: 0x18

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
outepint15 RO 0x0	outepint14 RO 0x0	outepint13 RO 0x0	outepint12 RO 0x0	outepint11 RO 0x0	outepint10 RO 0x0	outepint9 RO 0x0	outepint8 RO 0x0	outepint7 RO 0x0	outepint6 RO 0x0	outepint5 RO 0x0	outepint4 RO 0x0	outepint3 RO 0x0	outepint2 RO 0x0	outepint1 RO 0x0	outepint0 RO 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
inepint15 RO 0x0	inepint14 RO 0x0	inepint13 RO 0x0	inepint12 RO 0x0	inepint11 RO 0x0	inepint10 RO 0x0	inepint9 RO 0x0	inepint8 RO 0x0	inepint7 RO 0x0	inepint6 RO 0x0	inepint5 RO 0x0	inepint4 RO 0x0	inepint3 RO 0x0	inepint2 RO 0x0	inepint1 RO 0x0	inepint0 RO 0x0

daint Fields

Bit	Name	Description	Access	Reset
31	outepint15	OUT Endpoint 15 Interrupt Bit Value: Description 0x0: INACTIVE 0x1: ACTIVE	RO	0x0
30	outepint14	OUT Endpoint 14 Interrupt Bit Value: Description 0x0: INACTIVE 0x1: ACTIVE	RO	0x0
29	outepint13	OUT Endpoint 13 Interrupt Bit Value: Description 0x0: INACTIVE 0x1: ACTIVE	RO	0x0
28	outepint12	OUT Endpoint 12 Interrupt Bit Value: Description 0x0: INACTIVE 0x1: ACTIVE	RO	0x0
27	outepint11	OUT Endpoint 11 Interrupt Bit Value: Description 0x0: INACTIVE 0x1: ACTIVE	RO	0x0
26	outepint10	OUT Endpoint 10 Interrupt Bit Value: Description 0x0: INACTIVE 0x1: ACTIVE	RO	0x0

Bit	Name	Description	Access	Reset
25	outepint9	OUT Endpoint 9 Interrupt Bit **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0
24	outepint8	OUT Endpoint 8 Interrupt Bit **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0
23	outepint7	OUT Endpoint 7 Interrupt Bit **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0
22	outepint6	OUT Endpoint 6 Interrupt Bit **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0
21	outepint5	OUT Endpoint 5 Interrupt Bit **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0
20	outepint4	OUT Endpoint 4 Interrupt Bit **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0

Bit	Name	Description	Access	Reset
19	outepint3	OUT Endpoint 3 Interrupt Bit Value: Description 0x0: INACTIVE 0x1: ACTIVE	RO	0x0
18	outepint2	OUT Endpoint 2 Interrupt Bit Value: Description 0x0: INACTIVE 0x1: ACTIVE	RO	0x0
17	outepint1	OUT Endpoint 1 Interrupt Bit Value: Description 0x0: INACTIVE 0x1: ACTIVE	RO	0x0
16	outepint0	OUT Endpoint 0 Interrupt Bit Value: Description 0x0: INACTIVE 0x1: ACTIVE	RO	0x0
15	inepint15	IN Endpoint 15 Interrupt Bit Value: Description 0x0: INACTIVE 0x1: ACTIVE	RO	0x0
14	inepint14	IN Endpoint 14 Interrupt Bit Value: Description 0x0: INACTIVE 0x1: ACTIVE	RO	0x0

Bit	Name	Description	Access	Reset
13	inepint13	IN Endpoint 13 Interrupt Bit **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0
12	inepint12	IN Endpoint 12 Interrupt Bit **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0
11	inepint11	IN Endpoint 11 Interrupt Bit **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0
10	inepint10	IN Endpoint 10 Interrupt Bit **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0
9	inepint9	IN Endpoint 9 Interrupt Bit **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0
8	inepint8	IN Endpoint 8 Interrupt Bit **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0

Bit	Name	Description	Access	Reset
7	inepint7	IN Endpoint 7 Interrupt Bit Value: Description 0x0: INACTIVE 0x1: ACTIVE	RO	0x0
6	inepint6	IN Endpoint 6 Interrupt Bit Value: Description 0x0: INACTIVE 0x1: ACTIVE	RO	0x0
5	inepint5	IN Endpoint 5 Interrupt Bit Value: Description 0x0: INACTIVE 0x1: ACTIVE	RO	0x0
4	inepint4	IN Endpoint 4 Interrupt Bit Value: Description 0x0: INACTIVE 0x1: ACTIVE	RO	0x0
3	inepint3	IN Endpoint 3 Interrupt Bit Value: Description 0x0: INACTIVE 0x1: ACTIVE	RO	0x0
2	inepint2	IN Endpoint 2 Interrupt Bit Value: Description 0x0: INACTIVE 0x1: ACTIVE	RO	0x0

Bit	Name	Description	Access	Reset
1	inepint1	IN Endpoint 1 Interrupt Bit Value: Description 0x0: INACTIVE 0x1: ACTIVE	RO	0x0
0	inepint0	IN Endpoint 0 Interrupt Bit Value: Description 0x0: INACTIVE 0x1: ACTIVE	RO	0x0

daintmsk

Device All Endpoints Interrupt Mask Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB0081C
i_usbotg_1_devgrp	0xFFB40800	0xFFB4081C

Offset: 0x1C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
outepmsk15 RW 0x0	outepmsk14 RW 0x0	outepmsk13 RW 0x0	outepmsk12 RW 0x0	outepmsk11 RW 0x0	outepmsk10 RW 0x0	outepmsk9 RW 0x0	outepmsk8 RW 0x0	outepmsk7 RW 0x0	outepmsk6 RW 0x0	outepmsk5 RW 0x0	outepmsk4 RW 0x0	outepmsk3 RW 0x0	outepmsk2 RW 0x0	outepmsk1 RW 0x0	outepmsk0 RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
inepmsk15 RW 0x0	inepmsk14 RW 0x0	inepmsk13 RW 0x0	inepmsk12 RW 0x0	inepmsk11 RW 0x0	inepmsk10 RW 0x0	inepmsk9 RW 0x0	inepmsk8 RW 0x0	inepmsk7 RW 0x0	inepmsk6 RW 0x0	inepmsk5 RW 0x0	inepmsk4 RW 0x0	inepmsk3 RW 0x0	inepmsk2 RW 0x0	inepmsk1 RW 0x0	inepmsk0 RW 0x0

daintmsk Fields

Bit	Name	Description	Access	Reset
31	outepmsk15	OUT Endpoint 15 Interrupt mask Bit Value: Description 0x0: MASK 0x1: NOMASK	RW	0x0
30	outepmsk14	OUT Endpoint 14 Interrupt mask Bit Value: Description 0x0: MASK 0x1: NOMASK	RW	0x0
29	outepmsk13	OUT Endpoint 13 Interrupt mask Bit Value: Description 0x0: MASK 0x1: NOMASK	RW	0x0
28	outepmsk12	OUT Endpoint 12 Interrupt mask Bit Value: Description 0x0: MASK 0x1: NOMASK	RW	0x0
27	outepmsk11	OUT Endpoint 11 Interrupt mask Bit Value: Description 0x0: MASK 0x1: NOMASK	RW	0x0
26	outepmsk10	OUT Endpoint 10 Interrupt mask Bit Value: Description 0x0: MASK 0x1: NOMASK	RW	0x0

Bit	Name	Description	Access	Reset
25	outepmsk9	OUT Endpoint 9 Interrupt mask Bit Value: Description 0x0: MASK 0x1: NOMASK	RW	0x0
24	outepmsk8	OUT Endpoint 8 Interrupt mask Bit Value: Description 0x0: MASK 0x1: NOMASK	RW	0x0
23	outepmsk7	OUT Endpoint 7 Interrupt mask Bit Value: Description 0x0: MASK 0x1: NOMASK	RW	0x0
22	outepmsk6	OUT Endpoint 6 Interrupt mask Bit Value: Description 0x0: MASK 0x1: NOMASK	RW	0x0
21	outepmsk5	OUT Endpoint 5 Interrupt mask Bit Value: Description 0x0: MASK 0x1: NOMASK	RW	0x0
20	outepmsk4	OUT Endpoint 4 Interrupt mask Bit Value: Description 0x0: MASK 0x1: NOMASK	RW	0x0

Bit	Name	Description	Access	Reset
19	outepmsk3	OUT Endpoint 3 Interrupt mask Bit Value: Description 0x0: MASK 0x1: NOMASK	RW	0x0
18	outepmsk2	OUT Endpoint 2 Interrupt mask Bit Value: Description 0x0: MASK 0x1: NOMASK	RW	0x0
17	outepmsk1	OUT Endpoint 1 Interrupt mask Bit Value: Description 0x0: MASK 0x1: NOMASK	RW	0x0
16	outepmsk0	OUT Endpoint 0 Interrupt mask Bit Value: Description 0x0: MASK 0x1: NOMASK	RW	0x0
15	inepmsk15	IN Endpoint 15 Interrupt mask Bit Value: Description 0x0: INACTIVE 0x1: ACTIVE	RW	0x0
14	inepmsk14	IN Endpoint 14 Interrupt mask Bit Value: Description 0x0: MASK 0x1: NOMASK	RW	0x0

Bit	Name	Description	Access	Reset
13	inepmsk13	IN Endpoint 13 Interrupt mask Bit Value: Description 0x0: MASK 0x1: NOMASK	RW	0x0
12	inepmsk12	IN Endpoint 12 Interrupt mask Bit Value: Description 0x0: MASK 0x1: NOMASK	RW	0x0
11	inepmsk11	IN Endpoint 11 Interrupt mask Bit Value: Description 0x0: MASK 0x1: NOMASK	RW	0x0
10	inepmsk10	IN Endpoint 10 Interrupt mask Bit Value: Description 0x0: MASK 0x1: NOMASK	RW	0x0
9	inepmsk9	IN Endpoint 9 Interrupt mask Bit Value: Description 0x0: MASK 0x1: NOMASK	RW	0x0
8	inepmsk8	IN Endpoint 8 Interrupt mask Bit Value: Description 0x0: MASK 0x1: NOMASK	RW	0x0

Bit	Name	Description	Access	Reset
7	inepmsk7	IN Endpoint 7 Interrupt mask Bit Value: Description 0x0: MASK 0x1: NOMASK	RW	0x0
6	inepmsk6	IN Endpoint 6 Interrupt mask Bit Value: Description 0x0: MASK 0x1: NOMASK	RW	0x0
5	inepmsk5	IN Endpoint 5 Interrupt mask Bit Value: Description 0x0: MASK 0x1: NOMASK	RW	0x0
4	inepmsk4	IN Endpoint 4 Interrupt mask Bit Value: Description 0x0: MASK 0x1: NOMASK	RW	0x0
3	inepmsk3	IN Endpoint 3 Interrupt mask Bit Value: Description 0x0: MASK 0x1: NOMASK	RW	0x0
2	inepmsk2	IN Endpoint 2 Interrupt mask Bit Value: Description 0x0: MASK 0x1: NOMASK	RW	0x0

Bit	Name	Description	Access	Reset
1	inepmsk1	IN Endpoint 1 Interrupt mask Bit Value: Description 0x0: MASK 0x1: NOMASK	RW	0x0
0	inepmsk0	IN Endpoint 0 Interrupt mask Bit Value: Description 0x0: MASK 0x1: NOMASK	RW	0x0

dvbusdis

Device VBUS Discharge Time Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00828
i_usbotg_1_devgrp	0xFFB40800	0xFFB40828

Offset: 0x28

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
dvbusdis RW 0x17D7															

dvbusdis Fields

Bit	Name	Description	Access	Reset
15:0	dvbusdis	Device VBUS Discharge Time (DVBUSDis) Specifies the VBUS discharge time after VBUS pulsing during SRP. This value equals: VBUS discharge time in PHY clocks / 1, 024 The value you use depends whether the PHY is operating at 30 MHz (16-bit data width) or 60 MHz (8-bit data width). Depending on your VBUS load, this value can need adjustment.	RW	0x17D7

dvbuspulse

Device VBUS Pulsing Time Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB0082C
i_usbotg_1_devgrp	0xFFB40800	0xFFB4082C

Offset: 0x2C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				dvbuspulse RW 0x5B8											

dvbuspulse Fields

Bit	Name	Description	Access	Reset
11:0	dvbuspulse	Device VBUS Pulsing Time (DVBUSPulse) Specifies the VBUS pulsing time during SRP. This value equals: VBUS pulsing time in PHY clocks / 1, 024 The value you use depends whether the PHY is operating at 30 MHz (16-bit data width) or 60 MHz (8-bit data width).	RW	0x5B8

dthrctl

Device Threshold Control Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00830
i_usbotg_1_devgrp	0xFFB40800	0xFFB40830

Offset: 0x30

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				arbprken RW 0x1	Reserved	rxthrlen RW 0x8									rxthren RW 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			ahbthrratio RW 0x0		txthrlen RW 0x8									isothren RW 0x0	nonisothren RW 0x0

dthrctl Fields

Bit	Name	Description	Access	Reset
27	arbprken	Arbiter Parking Enable (ArbPrkEn) This bit controls internal DMA arbiter parking For IN endpoints. When thresholding is enabled and this bit is Set to one, Then the arbiter parks on the IN endpoint For which there is a token received on the USB. This is done to avoid getting into underrun conditions. By Default the parking is enabled. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x1
25:17	rxthrlen	Receive Threshold Length (RxThrLen) This field specifies Receive thresholding size in DWORDS. This field also specifies the amount of data received on the USB before the core can start transmitting on the AHB. The threshold length has to be at least eight DWORDS. The recommended value For ThrLen is to be the same as the programmed AHB Burst Length (GAHBCFG.HBstLen).	RW	0x8

Bit	Name	Description	Access	Reset
16	rxthren	Receive Threshold Enable (RxThrEn) When this bit is Set, the core enables thresholding in the receive direction. Note: We recommends that you do not enable RxThrEn, because it may cause issues in the RxFIFO especially during error conditions such as RxError and Babble. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
12:11	ahbthrratio	AHB Threshold Ratio (AHBThrRatio) These bits define the ratio between the AHB threshold and the MAC threshold for the transmit path only. The AHB threshold always remains less than or equal to the USB threshold, because this does not increase overhead. Both the AHB and the MAC threshold must be DWORD-aligned. The application needs to program TxThrLen and the AHBThrRatio to make the AHB Threshold value DWORD aligned. If the AHB threshold value is not DWORD aligned, the core might not behave correctly. When programming the TxThrLen and AHBThrRatio, the application must ensure that the minimum AHB threshold value does not go below 8 DWORDS to meet the USB turnaround time requirements. 2'b00: AHB threshold = MAC threshold 2'b01: AHB threshold = MAC threshold / 2 2'b10: AHB threshold = MAC threshold / 4 2'b11: AHB threshold = MAC threshold / 8 **Value** **Description** 0x0 THRESZERO 0x1 THRESONE 0x2 THRESTWO 0x3 THRESTHREE	RW	0x0

Bit	Name	Description	Access	Reset
10:2	txthrlen	Transmit Threshold Length (TxThrLen) This field specifies Transmit thresholding size in DWORDS. This also forms the MAC threshold and specifies the amount of data in bytes to be in the corresponding endpoint transmit FIFO, before the core can start transmit on the USB. The threshold length has to be at least eight DWORDS when the value of AHBThrRatio is 2'h00. In case the AHBThrRatio is non zero the application needs to ensure that the AHB Threshold value does not go below the recommended eight DWORD. This field controls both isochronous and non-isochronous IN endpoint thresholds. The recommended value for ThrLen is to be the same as the programmed AHB Burst Length (GAHBCFG.HBstLen).	RW	0x8
1	isothren	ISO IN Endpoints Threshold Enable. (ISOThrEn) When this bit is Set, the core enables thresholding For isochronous IN endpoints. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
0	nonisothren	Non-ISO IN Endpoints Threshold Enable. (NonISOThrEn) When this bit is Set, the core enables thresholding For Non Isochronous IN endpoints. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0

diepempmsk

Device IN Endpoint FIFO Empty Interrupt Mask Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00834
i_usbotg_1_devgrp	0xFFB40800	0xFFB40834

Offset: 0x34

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ineptxfempmsk15 RW 0x0	ineptxfempmsk14 RW 0x0	ineptxfempmsk13 RW 0x0	ineptxfempmsk12 RW 0x0	ineptxfempmsk11 RW 0x0	ineptxfempmsk10 RW 0x0	ineptxfempmsk9 RW 0x0	ineptxfempmsk8 RW 0x0	ineptxfempmsk7 RW 0x0	ineptxfempmsk6 RW 0x0	ineptxfempmsk5 RW 0x0	ineptxfempmsk4 RW 0x0	ineptxfempmsk3 RW 0x0	ineptxfempmsk2 RW 0x0	ineptxfempmsk1 RW 0x0	ineptxfempmsk0 RW 0x0

diepempmsk Fields

Bit	Name	Description	Access	Reset
15	ineptxfempmsk15	This bit acts as mask bits for DIEPINT15. Value: Description 0x0: MASK 0x1: NOMASK	RW	0x0
14	ineptxfempmsk14	This bit acts as mask bits for DIEPINT14. Value: Description 0x0: MASK 0x1: NOMASK	RW	0x0
13	ineptxfempmsk13	This bit acts as mask bits for DIEPINT13. Value: Description 0x0: MASK 0x1: NOMASK	RW	0x0
12	ineptxfempmsk12	This bit acts as mask bits for DIEPINT12. Value: Description 0x0: MASK 0x1: NOMASK	RW	0x0
11	ineptxfempmsk11	This bit acts as mask bits for DIEPINT11. Value: Description 0x0: MASK 0x1: NOMASK	RW	0x0

Bit	Name	Description	Access	Reset
10	ineptxfempmsk10	This bit acts as mask bits for DIEPINT10. **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0
9	ineptxfempmsk9	This bit acts as mask bits for DIEPINT9. **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0
8	ineptxfempmsk8	This bit acts as mask bits for DIEPINT8. **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0
7	ineptxfempmsk7	This bit acts as mask bits for DIEPINT7. **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0
6	ineptxfempmsk6	This bit acts as mask bits for DIEPINT6. **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0
5	ineptxfempmsk5	This bit acts as mask bits for DIEPINT5. **Value** **Description** 0x0 MASK 0x1 NOMASK	RW	0x0

Bit	Name	Description	Access	Reset
4	ineptxfempmsk4	This bit acts as mask bits for DIEPINT4. Value: Description 0x0: MASK 0x1: NOMASK	RW	0x0
3	ineptxfempmsk3	This bit acts as mask bits for DIEPINT3. Value: Description 0x0: MASK 0x1: NOMASK	RW	0x0
2	ineptxfempmsk2	This bit acts as mask bits for DIEPINT2. Value: Description 0x0: MASK 0x1: NOMASK	RW	0x0
1	ineptxfempmsk1	This bit acts as mask bits for DIEPINT1. Value: Description 0x0: MASK 0x1: NOMASK	RW	0x0
0	ineptxfempmsk0	This bit acts as mask bits for DIEPINT0. Value: Description 0x0: MASK 0x1: NOMASK	RW	0x0

diepctl0

Device Control IN Endpoint 0 Control Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00900
i_usbotg_1_devgrp	0xFFB40800	0xFFB40900

Offset: 0x100

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
epena RW 0x0	epdis RW 0x0	Reserved		snak WO 0x0	cnak WO 0x0	txfnum RW 0x0				stall RW 0x0	Reserved	eptype RO 0x0		naksts RO 0x0	Reserved
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
usbactep RO 0x1	Reserved													mps RW 0x0	

diepctl0 Fields

Bit	Name	Description	Access	Reset
31	epena	Endpoint Enable (EPEna) When Scatter/Gather DMA mode is enabled, For IN endpoints this bit indicates that the descriptor structure and data buffer with data ready to transmit is setup. When Scatter/Gather DMA mode is disabled such as in buffer pointer based DMA mode this bit indicates that data is ready to be transmitted on the endpoint. The core clears this bit before setting the following interrupts on this endpoint: Endpoint Disabled Transfer Completed **Value: Description** 0x0: INACTIVE 0x1: ACTIVE	RW	0x0
30	epdis	Endpoint Disable (EPDis) The application sets this bit to stop transmitting data on an endpoint, even before the transfer For that endpoint is complete. The applica-tion must wait For the Endpoint Disabled interrupt before treating the endpoint as disabled. The core clears this bit before setting the Endpoint Disabled Interrupt. The application must Set this bit only If Endpoint Enable is already Set For this endpoint. **Value: Description** 0x0: INACTIVE 0x1: ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
27	snak	Set NAK (SNAK) A write to this bit sets the NAK bit For the endpoint. Using this bit, the application can control the transmission of NAK handshakes on an endpoint. The core can also Set this bit For an endpoint after a SETUP packet is received on that endpoint. **Value** **Description** 0x0 NOSET 0x1 SET	WO	0x0
26	cnak	Clear NAK (CNAK) A write to this bit clears the NAK bit For the endpoint. **Value** **Description** 0x0 NOCLEAR 0x1 CLEAR	WO	0x0
25:22	txfnum	TxFIFO Number (TxFNum) For Shared FIFO operation, this value is always Set to 0, indicating that control IN endpoint 0 data is always written in the Non-Periodic Transmit FIFO. For Dedicated FIFO operation, this value is Set to the FIFO number that is assigned to IN Endpoint 0.	RW	0x0

Bit	Name	Description	Access	Reset
21	stall	STALL Handshake (Stall) The application can only Set this bit, and the core clears it, when a SETUP token is received For this endpoint. If a NAK bit, Global Nonperiodic IN NAK, or Global OUT NAK is Set along with this bit, the STALL bit takes priority. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
19:18	eptype	Endpoint Type (EPType) Hardcoded to 00 For control. **Value** **Description** 0x0 ACTIVE	RO	0x0
17	naksts	NAK Status (NAKSts) Indicates the following: 1'b0: The core is transmitting non-NAK handshakes based on the FIFO status 1'b1: The core is transmitting NAK handshakes on this endpoint. When this bit is Set, either by the application or core, the core stops transmitting data, even If there is data available in the TxFIFO. Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0

Bit	Name	Description	Access	Reset
15	usbactep	USB Active Endpoint (USBActEP) This bit is always SET to 1, indicating that control endpoint 0 is always active in all configurations and interfaces. **Value** **Description** 0x1 ACTIVE0	RO	0x1
1:0	mps	Maximum Packet Size (MPS) Applies to IN and OUT endpoints. The application must program this field with the maximum packet size For the current logical endpoint. 2'b00: 64 bytes 2'b01: 32 bytes 2'b10: 16 bytes 2'b11: 8 bytes **Value** **Description** 0x0 BYTES64 0x1 BYTES32 0x2 BYTES16 0x3 BYTES8	RW	0x0

diepint0

Device IN Endpoint 0 Interrupt Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00908
i_usbotg_1_devgrp	0xFFB40800	0xFFB40908

Offset: 0x108

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	txfifoundrn RW 0x0	txfempp RO 0x1	inepnakeff RW 0x0	intknepmis RW 0x0	intkntxfempp RW 0x0	timeout RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0

diepint0 Fields

Bit	Name	Description	Access	Reset
14	nyetintrpt	NYET Interrupt (NYETIntrpt) The core generates this interrupt when a NYET response is transmitted for a non isochronous OUT endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
13	nakintrpt	NAK Interrupt (NAKInterrupt) The core generates this interrupt when a NAK is transmitted or received by the device. In case of isochronous IN endpoints the interrupt gets generated when a zero length packet is transmitted due to un-availability of data in the TXFifo. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
12	bbleerr	NAK Interrupt (BbleErr) The core generates this interrupt when babble is received for the endpoint. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
11	pktdrpsts	Packet Drop Status (PktDrpSts) This bit indicates to the application that an ISOC OUT packet has been dropped. This bit does not have an associated mask bit and does not generate an interrupt. Dependency: This bit is valid in non Scatter/Gather DMA mode when periodic transfer interrupt feature is selected. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
9	bnaintr	BNA (Buffer Not Available) Interrupt (BNAIntr) This bit is valid only when Scatter/Gather DMA mode is enabled. The core generates this interrupt when the descriptor accessed is not ready For the Core to process, such as Host busy or DMA done **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
8	txfifoundrn	Fifo Underrun (TxfifoUndrn) Applies to IN endpoints Only The core generates this interrupt when it detects a transmit FIFO underrun condition in threshold mode For this endpoint. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
7	txfemp	Transmit FIFO Empty (TxFEmp) This bit is valid only For IN Endpoints This interrupt is asserted when the TxFIFO For this endpoint is either half or completely empty. The half or completely empty status is determined by the TxFIFO Empty Level bit in the Core AHB Configuration register (GAHBCFG.NPTxFEmpLvl)). **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x1

Bit	Name	Description	Access	Reset
6	inepnakeff	IN Endpoint NAK Effective (INEPNakEff) Applies to periodic IN endpoints only. This bit can be cleared when the application clears the IN endpoint NAK by writing to DIEPCTLn.CNAK. This interrupt indicates that the core has sampled the NAK bit Set (either by the application or by the core). The interrupt indicates that the IN endpoint NAK bit Set by the application has taken effect in the core. This interrupt does not guarantee that a NAK handshake is sent on the USB. A STALL bit takes priority over a NAK bit. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
5	intknepmis	IN Token Received with EP Mismatch (INTknEPMis) Applies to non-periodic IN endpoints only. Indicates that the data in the top of the non-periodic TxFIFO belongs to an endpoint other than the one For which the IN token was received. This interrupt is asserted on the endpoint For which the IN token was received. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
4	intkntxfemp	IN Token Received When TxFIFO is Empty (INTknTXFEmp) Applies to non-periodic IN endpoints only. Indicates that an IN token was received when the associated TxFIFO (periodic/non-periodic) was empty. This interrupt is asserted on the endpoint For which the IN token was received. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
3	timeout	Timeout Condition (TimeOUT) In shared TX FIFO mode, applies to non-isochronous IN endpoints only. In dedicated FIFO mode, applies only to Control IN endpoints. In Scatter/Gather DMA mode, the TimeOUT interrupt is not asserted. Indicates that the core has detected a timeout condition on the USB For the last IN token on this endpoint. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
2	ahberr	AHB Error (AHBErr) Applies to IN and OUT endpoints. This is generated only in Internal DMA mode when there is an AHB error during an AHB read/write. The application can read the corresponding endpoint DMA address register to get the error address. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
1	epdisbld	Endpoint Disabled Interrupt (EPDisbld) Applies to IN and OUT endpoints. This bit indicates that the endpoint is disabled per the application's request. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
0	xfercompl	Transfer Completed Interrupt (XferCompl) Applies to IN and OUT endpoints. When Scatter/Gather DMA mode is enabled - For IN endpoint this field indicates that the requested data from the descriptor is moved from external system memory to internal FIFO. - For OUT endpoint this field indicates that the requested data from the internal FIFO is moved to external system memory. This interrupt is generated only when the corresponding endpoint descriptor is closed, and the IOC bit For the corresponding descriptor is Set. When Scatter/Gather DMA mode is disabled, this field indicates that the programmed transfer is complete on the AHB as well as on the USB, For this endpoint. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

dieptsiz0

Device IN Endpoint 0 Transfer Size Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00910
i_usbotg_1_devgrp	0xFFB40800	0xFFB40910

Offset: 0x110

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved											pktcnt RW 0x0		Reserved		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									xfersize RW 0x0						

dieptsiz0 Fields

Bit	Name	Description	Access	Reset
20:19	pktcnt	Packet Count (PktCnt) Indicates the total number of USB packets that constitute the Transfer Size amount of data For endpoint 0. In Endpoints : This field is decremented every time a packet (maximum size or short packet) is read from the TxFIFO. OUT Endpoints: This field is decremented every time a packet (maximum size or short packet) is written to the RxFIFO.	RW	0x0

Bit	Name	Description	Access	Reset
6:0	xfersize	Transfer Size (XferSize) This field contains the transfer size in bytes for the current endpoint. The transfer size (XferSize) = Sum of buffer sizes across all descriptors in the list for the endpoint. In Buffer DMA, the core only interrupts the application after it has exhausted the transfer size amount of data. The transfer size can be set to the maximum packet size of the endpoint, to be interrupted at the end of each packet. IN Endpoints: The core decrements this field every time a packet from the external memory is written to the TxFIFO. OUT Endpoints: The core decrements this field every time a packet is read from the RxFIFO and written to the external memory.	RW	0x0

diepdma0

Device IN Endpoint 0 DMA Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00914
i_usbotg_1_devgrp	0xFFB40800	0xFFB40914

Offset: 0x114

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepdma0 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
diepdma0 RW 0x0															

diepdma0 Fields

Bit	Name	Description	Access	Reset
31:0	diepdma0	Holds the start address of the external memory for storing or fetching endpoint data. Note: For control endpoints, this field stores control OUT data packets as well as SETUP transaction data packets. When more than three SETUP packets are received back-to-back, the SETUP data packet in the memory is overwritten. This register is incremented on every AHB transaction. The application can give only a DWORD-aligned address. When Scatter/Gather DMA mode is not enabled, the application programs the start address value in this field. When Scatter/Gather DMA mode is enabled, this field indicates the base pointer for the descriptor list.	RW	0x0

dtxfsts0

Device IN Endpoint Transmit FIFO Status Register 0

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00918
i_usbotg_1_devgrp	0xFFB40800	0xFFB40918

Offset: 0x118

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ineptxfspcavail RO 0x2000															

dtxfsts0 Fields

Bit	Name	Description	Access	Reset
15:0	ineptxfspcavail	IN Endpoint TxFIFO Space Avail (INEPTxFSpcAvail) Indicates the amount of free space available in the Endpoint TxFIFO. Values are in terms of 32-bit words. 16'h0: Endpoint TxFIFO is full 16'h1: 1 word available 16'h2: 2 words available 16'hn: n words available (where 0 n 32,768) 16'h8000: 32,768 words available Others: Reserved	RO	0x2000

diepdmab0

Device IN Endpoint 16 Buffer Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB0091C
i_usbotg_1_devgrp	0xFFB40800	0xFFB4091C

Offset: 0x11C

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepdmab0 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
diepdmab0 RO 0x0															

diepdmab0 Fields

Bit	Name	Description	Access	Reset
31:0	diepdmab0	Holds the current buffer address.This register is updated as and when the data transfer for the corresponding end point is in progress. This register is present only in Scatter/Gather DMA mode. Otherwise this field is reserved.	RO	0x0

diepctl1

Device Control IN Endpoint 1 Control Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00920
i_usbotg_1_devgrp	0xFFB40800	0xFFB40920

Offset: 0x120

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	txfnum RW 0x0				stall RW 0x0	Reserved	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
usbactep RW 0x0	Reserved				mps RW 0x0										

diepctl1 Fields

Bit	Name	Description	Access	Reset
31	epena	Endpoint Enable (EPEna) Applies to IN and OUT endpoints. When Scatter/Gather DMA mode is enabled, For IN endpoints this bit indicates that the descriptor structure and data buffer with data ready to transmit is setup. For OUT endpoint it indicates that the descriptor structure and data buffer to receive data is setup. When Scatter/Gather DMA mode is enabledsuch as for buffer-pointer based DMA mode: - For IN endpoints, this bit indicates that data is ready to be transmitted on the endpoint. - For OUT endpoints, this bit indicates that the application has allocated the memory to start receiving data from the USB. - The core clears this bit before setting any of the following interrupts on this endpoint: SETUP Phase Done Endpoint Disabled Transfer Completed Note: For control endpoints in DMA mode, this bit must be set to be able to transfer SETUP data packets in memory. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
30	epdis	Endpoint Disable (EPDis) Applies to IN and OUT endpoints. The application sets this bit to stop transmitting/receiving data on an endpoint, even before the transfer for that endpoint is complete. The application must wait for the Endpoint Disabled interrupt before treating the endpoint as disabled. The core clears this bit before setting the Endpoint Disabled interrupt. The application must set this bit only if Endpoint Enable is already set for this endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
29	setd1pid	Set DATA1 PID (SetD1PID) Applies to interrupt/bulk IN and OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA1. This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. Set Odd (micro)frame (SetOddFr) Applies to isochronous IN and OUT endpoints only. Writing to this field sets the Even/Odd (micro)frame (EO_FrNum) field to odd (micro)frame. This field is not applicable for Scatter/Gather DMA mode. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	WO	0x0

Bit	Name	Description	Access	Reset
28	setd0pid	Set DATA0 PID (SetD0PID) Applies to interrupt/bulk IN and OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA0. This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. 1'b0 WO In non-Scatter/Gather DMA mode: Set Even (micro)frame (SetEvenFr) Applies to isochronous IN and OUT endpoints only. Writing to this field sets the Even/Odd (micro)frame (EO_FrNum) field to even (micro) frame. When Scatter/Gather DMA mode is enabled, this field is reserved. The frame number in which to send data is in the transmit descriptor structure. The frame in which to receive data is updated in receive descriptor structure. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	WO	0x0
27	snak	Set NAK (SNAK) A write to this bit sets the NAK bit For the endpoint. Using this bit, the application can control the transmission of NAK handshakes on an endpoint. The core can also Set this bit For an endpoint after a SETUP packet is received on that endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	WO	0x0

Bit	Name	Description	Access	Reset
26	cnak	Clear NAK (CNAK) A write to this bit clears the NAK bit For the endpoint. Value: 0x0 — INACTIVE Value: 0x1 — ACTIVE	WO	0x0
25:22	txfnum	TxFIFO Number (TxFNum) Shared FIFO Operation non-periodic endpoints must set this bit to zero. Periodic endpoints must map this to the corresponding Periodic TxFIFO number. 4'h0: Non-Periodic TxFIFO Others: Specified Periodic TxFIFO.number Note: An interrupt IN endpoint can be configured as a non-periodic endpoint for applications such as mass storage. The core treats an IN endpoint as a non-periodic endpoint if the TxFNum field is set to 0. Otherwise, a separate periodic FIFO must be allocated for an interrupt IN endpoint, and the number of this FIFO must be programmed into the TxFNum field. Configuring an interrupt IN endpoint as a non-periodic endpoint saves the extra periodic FIFO area. Dedicated FIFO Operationthese bits specify the FIFO number associated with this endpoint. Each active IN endpoint must be programmed to a separate FIFO number. This field is valid only for IN endpoints.	RW	0x0

Bit	Name	Description	Access	Reset
21	stall	STALL Handshake (Stall) Applies to non-control, non-isochronous IN and OUT endpoints only. The application sets this bit to stall all tokens from the USB host to this endpoint. If a NAK bit, Global Non-periodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority. Only the application can clear this bit, never the core. 1'b0 R_W Applies to control endpoints only. The application can only set this bit, and the core clears it, when a SETUP token is received for this endpoint. If a NAK bit, Global Non-periodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority. Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
19:18	eptype	Endpoint Type (EPType) This is the transfer type supported by this logical endpoint. 2'b00: Control 2'b01: Isochronous 2'b10: Bulk 2'b11: Interrupt **Value** **Description** 0x0 CONTROL 0x1 ISOCHRONOUS 0x2 BULK 0x3 INTERRUP	RW	0x0

Bit	Name	Description	Access	Reset
17	naksts	NAK Status (NAKSts) Indicates the following: 1'b0: The core is transmitting non-NAK handshakes based on the FIFO status. 1'b1: The core is transmitting NAK handshakes on this endpoint. When either the application or the core sets this bit: The core stops receiving any data on an OUT endpoint, even if there is space in the RxFIFO to accommodate the incoming packet. For non-isochronous IN endpoints: The core stops transmitting any data on an IN endpoint, even if there data is available in the TxFIFO. For isochronous IN endpoints: The core sends out a zero-length data packet, even if there data is available in the TxFIFO. Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake. **Value** — **Description** 0x0 — NONNAK 0x1 — NAK	RO	0x0

Bit	Name	Description	Access	Reset
16	dpid	Endpoint Data PID (DPID) Applies to interrupt/bulk IN and OUT endpoints only. Contains the PID of the packet to be received or transmitted on this endpoint. The application must program the PID of the first packet to be received or transmitted on this endpoint, after the endpoint is activated. The applications use the SetD1PID and SetD0PID fields of this register to program either DATA0 or DATA1 PID. 1'b0: DATA0 1'b1: DATA1 This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. 1'b0 RO Even/Odd (Micro)Frame (EO_FrNum) In non-Scatter/Gather DMA mode: Applies to isochronous IN and OUT endpoints only. Indicates the (micro)frame number in which the core transmits/receives isochronous data for this endpoint. The application must program the even/odd (micro) frame number in which it intends to transmit/receive isochronous data for this endpoint using the SetEvnFr and SetOddFr fields in this register. 1'b0: Even (micro)frame 1'b1: Odd (micro)frame When Scatter/Gather DMA mode is enabled, this field is reserved. The frame number in which to send data is provided in the transmit descriptor structure. The frame in which data is received is updated in receive descriptor structure. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RO	0x0

Bit	Name	Description	Access	Reset
15	usbactep	USB Active Endpoint (USBActEP) Indicates whether this endpoint is active in the current configuration and interface. The core clears this bit for all endpoints (other than EP 0) after detecting a USB reset. After receiving the SetConfiguration and SetInterface commands, the applica-tion must program endpoint registers accordingly and set this bit. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
10:0	mps	Maximum Packet Size (MPS) The application must program this field with the maximum packet size for the current logical endpoint. This value is in bytes.	RW	0x0

diepint1

Device IN Endpoint 1 Interrupt Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00928
i_usbotg_1_devgrp	0xFFB40800	0xFFB40928

Offset: 0x128

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	txfifoundrn RW 0x0	txfempp RO 0x1	inepnakeff RW 0x0	intknepmis RW 0x0	intkntxfempp RW 0x0	timeout RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0

diepint1 Fields

Bit	Name	Description	Access	Reset
14	nyetintrpt	NYET Interrupt (NYETIntrpt) The core generates this interrupt when a NYET response is transmitted for a non isochronous OUT endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
13	nakintrpt	NAK Interrupt (NAKInterrupt) The core generates this interrupt when a NAK is transmitted or received by the device. In case of isochronous IN endpoints the interrupt gets generated when a zero length packet is transmitted due to un-availability of data in the TXFifo. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
12	bbleerr	NAK Interrupt (BbleErr) The core generates this interrupt when babble is received for the endpoint. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
11	pktdrpsts	Packet Drop Status (PktDrpSts) This bit indicates to the application that an ISOC OUT packet has been dropped. This bit does not have an associated mask bit and does not generate an interrupt. Dependency: This bit is valid in non Scatter/Gather DMA mode when periodic transfer interrupt feature is selected. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
9	bnaintr	BNA (Buffer Not Available) Interrupt (BNAIntr) This bit is valid only when Scatter/Gather DMA mode is enabled. The core generates this interrupt when the descriptor accessed is not ready For the Core to process, such as Host busy or DMA done **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
8	txfifoundrn	Fifo Underrun (TxfifoUndrn) Applies to IN endpoints Only This bit is valid only If thresholding is enabled. The core generates this interrupt when it detects a transmit FIFO underrun condition For this endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
7	txfemp	Transmit FIFO Empty (TxFEmp) This bit is valid only For IN Endpoints This interrupt is asserted when the TxFIFO For this endpoint is either half or completely empty. The half or completely empty status is determined by the TxFIFO Empty Level bit in the Core AHB Configuration register (GAHBCFG.NPTxFEmpLvl)). **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RO	0x1

Bit	Name	Description	Access	Reset
6	inepnakeff	IN Endpoint NAK Effective (INEPNakEff) Applies to periodic IN endpoints only. This bit can be cleared when the application clears the IN endpoint NAK by writing to DIEPCTLn.CNAK. This interrupt indicates that the core has sampled the NAK bit Set (either by the application or by the core). The interrupt indicates that the IN endpoint NAK bit Set by the application has taken effect in the core. This interrupt does not guarantee that a NAK handshake is sent on the USB. A STALL bit takes priority over a NAK bit. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
5	intknepmis	IN Token Received with EP Mismatch (INTknEPMis) Applies to non-periodic IN endpoints only. Indicates that the data in the top of the non-periodic TxFIFO belongs to an endpoint other than the one For which the IN token was received. This interrupt is asserted on the endpoint For which the IN token was received. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
4	intkntxfemp	IN Token Received When TxFIFO is Empty (INTknTXFEmp) Applies to non-periodic IN endpoints only. Indicates that an IN token was received when the associated TxFIFO (periodic/non-periodic) was empty. This interrupt is asserted on the endpoint For which the IN token was received. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
3	timeout	Timeout Condition (TimeOUT) In shared TX FIFO mode, applies to non-isochronous IN endpoints only. In dedicated FIFO mode, applies only to Control IN endpoints. In Scatter/Gather DMA mode, the TimeOUT interrupt is not asserted. Indicates that the core has detected a timeout condition on the USB For the last IN token on this endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
2	ahberr	AHB Error (AHBErr) Applies to IN and OUT endpoints. This is generated only in Internal DMA mode when there is an AHB error during an AHB read/write. The application can read the corresponding endpoint DMA address register to get the error address. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
1	epdisbld	Endpoint Disabled Interrupt (EPDisbld) Applies to IN and OUT endpoints. This bit indicates that the endpoint is disabled per the application's request. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
0	xfercompl	Transfer Completed Interrupt (XferCompl) Applies to IN and OUT endpoints. When Scatter/Gather DMA mode is enabled - For IN endpoint this field indicates that the requested data from the descriptor is moved from external system memory to internal FIFO. - For OUT endpoint this field indicates that the requested data from the internal FIFO is moved to external system memory. This interrupt is generated only when the corresponding endpoint descriptor is closed, and the IOC bit For the corresponding descriptor is Set. When Scatter/Gather DMA mode is disabled, this field indicates that the programmed transfer is complete on the AHB as well as on the USB, For this endpoint. Value: Description 0x0: INACTIVE 0x1: ACTIVE	RW	0x0

dieptsiz1

Device IN Endpoint 1 Transfer Size Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00930
i_usbotg_1_devgrp	0xFFB40800	0xFFB40930

Offset: 0x130

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	mc RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
xfersize RW 0x0															

dieptsiz1 Fields

Bit	Name	Description	Access	Reset
30:29	mc	Applies to IN endpoints only. For periodic IN endpoints, this field indicates the number of packets that must be transmitted per microframe on the USB. The core uses this field to calculate the data PID for isochronous IN endpoints. 2'b01: 1 packet 2'b10: 2 packets 2'b11: 3 packets For non-periodic IN endpoints, this field is valid only in Internal DMA mode. It specifies the number of packets the core must fetchfor an IN endpoint before it switches to the endpoint pointed to by the Next Endpoint field of the Device Endpoint-n Control register (DIEPCTLn.NextEp) **Value** **Description** 0x1 PACKETONE 0x2 PACKETTWO 0x3 PACKETTHREE	RW	0x0

Bit	Name	Description	Access	Reset
28:19	pktcnt	Packet Count (PktCnt) Indicates the total number of USB packets that constitute the Transfer Size amount of data For endpoint 0. This field is decremented every time a packet (maximum size or short packet) is read from the TxFIFO.	RW	0x0
18:0	xfersize	Transfer Size (XferSize) Indicates the transfer size in bytes For endpoint 0. The core interrupts the application only after it has exhausted the transfer size amount of data. The transfer size can be Set to the maximum packet size of the endpoint, to be interrupted at the end of each packet. The core decrements this field every time a packet from the external memory is written to the TxFIFO.	RW	0x0

diepdma1

Device IN Endpoint 1 DMA Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00934
i_usbotg_1_devgrp	0xFFB40800	0xFFB40934

Offset: 0x134

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepdma1 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
diepdma1 RW 0x0															

diepdma1 Fields

Bit	Name	Description	Access	Reset
31:0	diepdma1	Holds the start address of the external memory for storing or fetching endpoint data. Note: For control endpoints, this field stores control OUT data packets as well as SETUP transaction data packets. When more than three SETUP packets are received back-to-back, the SETUP data packet in the memory is overwritten. This register is incremented on every AHB transaction. The application can give only a DWORD-aligned address. When Scatter/Gather DMA mode is not enabled, the application programs the start address value in this field. When Scatter/Gather DMA mode is enabled, this field indicates the base pointer for the descriptor list.	RW	0x0

dtxfsts1

Device IN Endpoint Transmit FIFO Status Register 1

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00938
i_usbotg_1_devgrp	0xFFB40800	0xFFB40938

Offset: 0x138

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ineptxfspcavail RO 0x2000															

dtxfsts1 Fields

Bit	Name	Description	Access	Reset
15:0	ineptxfspcavail	IN Endpoint TxFIFO Space Avail (INEPTxFSpcAvail) Indicates the amount of free space available in the Endpoint TxFIFO. Values are in terms of 32-bit words. 16'h0: Endpoint TxFIFO is full 16'h1: 1 word available 16'h2: 2 words available 16'hn: n words available (where 0 n 32,768) 16'h8000: 32,768 words available Others: Reserved	RO	0x2000

diepdmab1

Device IN Endpoint 1 Buffer Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB0093C
i_usbotg_1_devgrp	0xFFB40800	0xFFB4093C

Offset: 0x13C

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepdmab1 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
diepdmab1 RO 0x0															

diepdmab1 Fields

Bit	Name	Description	Access	Reset
31:0	diepdmab1	Holds the current buffer address.This register is updated as and when the data transfer for the corresponding end point is in progress. This register is present only in Scatter/Gather DMA mode. Otherwise this field is reserved.	RO	0x0

diepctl2

Device Control IN Endpoint 2 Control Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00940
i_usbotg_1_devgrp	0xFFB40800	0xFFB40940

Offset: 0x140

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	txfnum RW 0x0				stall RW 0x0	Reserved	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
usbactep RW 0x0	Reserved				mps RW 0x0										

diepctl2 Fields

Bit	Name	Description	Access	Reset
31	epena	Endpoint Enable (EPEna) Applies to IN and OUT endpoints. When Scatter/Gather DMA mode is enabled, For IN endpoints this bit indicates that the descriptor structure and data buffer with data ready to transmit is setup. For OUT endpoint it indicates that the descriptor structure and data buffer to receive data is setup. When Scatter/Gather DMA mode is enabledsuch as for buffer-pointer based DMA mode: - For IN endpoints, this bit indicates that data is ready to be transmitted on the endpoint. - For OUT endpoints, this bit indicates that the application has allocated the memory to start receiving data from the USB. - The core clears this bit before setting any of the following interrupts on this endpoint: SETUP Phase Done Endpoint Disabled Transfer Completed Note: For control endpoints in DMA mode, this bit must be set to be able to transfer SETUP data packets in memory. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
30	epdis	Endpoint Disable (EPDis) Applies to IN and OUT endpoints. The application sets this bit to stop transmitting/receiving data on an endpoint, even before the transfer for that endpoint is complete. The application must wait for the Endpoint Disabled interrupt before treating the endpoint as disabled. The core clears this bit before setting the Endpoint Disabled interrupt. The application must set this bit only if Endpoint Enable is already set for this endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
29	setd1pid	Set DATA1 PID (SetD1PID) Applies to interrupt/bulk IN and OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA1. This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. Set Odd (micro)frame (SetOddFr) Applies to isochronous IN and OUT endpoints only. Writing to this field sets the Even/Odd (micro)frame (EO_FrNum) field to odd (micro)frame. This field is not applicable for Scatter/Gather DMA mode. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	WO	0x0

Bit	Name	Description	Access	Reset
28	setd0pid	Set DATA0 PID (SetD0PID) Applies to interrupt/bulk IN and OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA0. This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. 1'b0 WO In non-Scatter/Gather DMA mode: Set Even (micro)frame (SetEvenFr) Applies to isochronous IN and OUT endpoints only. Writing to this field sets the Even/Odd (micro)frame (EO_FrNum) field to even (micro) frame. When Scatter/Gather DMA mode is enabled, this field is reserved. The frame number in which to send data is in the transmit descriptor structure. The frame in which to receive data is updated in receive descriptor structure. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	WO	0x0
27	snak	Set NAK (SNAK) A write to this bit sets the NAK bit For the endpoint. Using this bit, the application can control the transmission of NAK handshakes on an endpoint. The core can also Set this bit For an endpoint after a SETUP packet is received on that endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	WO	0x0

Bit	Name	Description	Access	Reset
26	cnak	Clear NAK (CNAK) A write to this bit clears the NAK bit For the endpoint. Value: 0x0 — INACTIVE Value: 0x1 — ACTIVE	WO	0x0
25:22	txfnum	TxFIFO Number (TxFNum) Shared FIFO Operation non-periodic endpoints must set this bit to zero. Periodic endpoints must map this to the corresponding Periodic TxFIFO number. 4'h0: Non-Periodic TxFIFO Others: Specified Periodic TxFIFO.number Note: An interrupt IN endpoint can be configured as a non-periodic endpoint for applications such as mass storage. The core treats an IN endpoint as a non-periodic endpoint if the TxFNum field is set to 0. Otherwise, a separate periodic FIFO must be allocated for an interrupt IN endpoint, and the number of this FIFO must be programmed into the TxFNum field. Configuring an interrupt IN endpoint as a non-periodic endpoint saves the extra periodic FIFO area. Dedicated FIFO Operationthese bits specify the FIFO number associated with this endpoint. Each active IN endpoint must be programmed to a separate FIFO number. This field is valid only for IN endpoints.	RW	0x0

Bit	Name	Description	Access	Reset
21	stall	STALL Handshake (Stall) Applies to non-control, non-isochronous IN and OUT endpoints only. The application sets this bit to stall all tokens from the USB host to this endpoint. If a NAK bit, Global Non-periodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority. Only the application can clear this bit, never the core. 1'b0 R_W Applies to control endpoints only. The application can only set this bit, and the core clears it, when a SETUP token is received for this endpoint. If a NAK bit, Global Non-periodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority. Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake. **Value: Description** 0x0: INACTIVE 0x1: ACTIVE	RW	0x0
19:18	eptype	Endpoint Type (EPType) This is the transfer type supported by this logical endpoint. 2'b00: Control 2'b01: Isochronous 2'b10: Bulk 2'b11: Interrupt **Value: Description** 0x0: CONTROL 0x1: ISOCHRONOUS 0x2: BULK 0x3: INTERRUP	RW	0x0

Bit	Name	Description	Access	Reset
17	naksts	NAK Status (NAKSts) Indicates the following: 1'b0: The core is transmitting non-NAK handshakes based on the FIFO status. 1'b1: The core is transmitting NAK handshakes on this endpoint. When either the application or the core sets this bit: The core stops receiving any data on an OUT endpoint, even if there is space in the RxFIFO to accommodate the incoming packet. For non-isochronous IN endpoints: The core stops transmitting any data on an IN endpoint, even if there data is available in the TxFIFO. For isochronous IN endpoints: The core sends out a zero-length data packet, even if there data is available in the TxFIFO. Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake. **Value** — **Description** 0x0 — NONNAK 0x1 — NAK	RO	0x0

Bit	Name	Description	Access	Reset
16	dpid	Endpoint Data PID (DPID) Applies to interrupt/bulk IN and OUT endpoints only. Contains the PID of the packet to be received or transmitted on this endpoint. The application must program the PID of the first packet to be received or transmitted on this endpoint, after the endpoint is activated. The applications use the SetD1PID and SetD0PID fields of this register to program either DATA0 or DATA1 PID. 1'b0: DATA0 1'b1: DATA1 This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. 1'b0 RO Even/Odd (Micro)Frame (EO_FrNum) In non-Scatter/Gather DMA mode: Applies to isochronous IN and OUT endpoints only. Indicates the (micro)frame number in which the core transmits/receives isochronous data for this endpoint. The application must program the even/odd (micro) frame number in which it intends to transmit/receive isochronous data for this endpoint using the SetEvnFr and SetOddFr fields in this register. 1'b0: Even (micro)frame 1'b1: Odd (micro)frame When Scatter/Gather DMA mode is enabled, this field is reserved. The frame number in which to send data is provided in the transmit descriptor structure. The frame in which data is received is updated in receive descriptor structure. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RO	0x0

Bit	Name	Description	Access	Reset
15	usbactep	USB Active Endpoint (USBActEP) Indicates whether this endpoint is active in the current configuration and interface. The core clears this bit for all endpoints (other than EP 0) after detecting a USB reset. After receiving the SetConfiguration and SetInterface commands, the applica-tion must program endpoint registers accordingly and set this bit. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
10:0	mps	Maximum Packet Size (MPS) The application must program this field with the maximum packet size for the current logical endpoint. This value is in bytes.	RW	0x0

diepint2

Device IN Endpoint 2 Interrupt Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00948
i_usbotg_1_devgrp	0xFFB40800	0xFFB40948

Offset: 0x148

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	txfifoundrn RW 0x0	txfempp RO 0x1	inepnakeff RW 0x0	intknepmis RW 0x0	intkntxfempp RW 0x0	timeout RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0

diepint2 Fields

Bit	Name	Description	Access	Reset
14	nyetintrpt	NYET Interrupt (NYETIntrpt) The core generates this interrupt when a NYET response is transmitted for a non isochronous OUT endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
13	nakintrpt	NAK Interrupt (NAKInterrupt) The core generates this interrupt when a NAK is transmitted or received by the device. In case of isochronous IN endpoints the interrupt gets generated when a zero length packet is transmitted due to un-availability of data in the TXFifo. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
12	bbleerr	NAK Interrupt (BbleErr) The core generates this interrupt when babble is received for the endpoint. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
11	pktdrpsts	Packet Drop Status (PktDrpSts) This bit indicates to the application that an ISOC OUT packet has been dropped. This bit does not have an associated mask bit and does not generate an interrupt. Dependency: This bit is valid in non Scatter/Gather DMA mode when periodic transfer interrupt feature is selected. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
9	bnaintr	BNA (Buffer Not Available) Interrupt (BNAIntr) This bit is valid only when Scatter/Gather DMA mode is enabled. The core generates this interrupt when the descriptor accessed is not ready For the Core to process, such as Host busy or DMA done **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
8	txfifoundrn	Fifo Underrun (TxfifoUndrn) Applies to IN endpoints Only This bit is valid only If thresholding is enabled. The core generates this interrupt when it detects a transmit FIFO underrun condition For this endpoint. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
7	txfemp	Transmit FIFO Empty (TxFEmp) This bit is valid only For IN Endpoints This interrupt is asserted when the TxFIFO For this endpoint is either half or completely empty. The half or completely empty status is determined by the TxFIFO Empty Level bit in the Core AHB Configuration register (GAHBCFG.NPTxFEmpLvl)). **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x1

Bit	Name	Description	Access	Reset
6	inepnakeff	IN Endpoint NAK Effective (INEPNakEff) Applies to periodic IN endpoints only. This bit can be cleared when the application clears the IN endpoint NAK by writing to DIEPCTLn.CNAK. This interrupt indicates that the core has sampled the NAK bit Set (either by the application or by the core). The interrupt indicates that the IN endpoint NAK bit Set by the application has taken effect in the core. This interrupt does not guarantee that a NAK handshake is sent on the USB. A STALL bit takes priority over a NAK bit. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
5	intknepmis	IN Token Received with EP Mismatch (INTknEPMis) Applies to non-periodic IN endpoints only. Indicates that the data in the top of the non-periodic TxFIFO belongs to an endpoint other than the one For which the IN token was received. This interrupt is asserted on the endpoint For which the IN token was received. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
4	intkntxfemp	IN Token Received When TxFIFO is Empty (INTknTXFEmp) Applies to non-periodic IN endpoints only. Indicates that an IN token was received when the associated TxFIFO (periodic/non-periodic) was empty. This interrupt is asserted on the endpoint For which the IN token was received. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
3	timeout	Timeout Condition (TimeOUT) In shared TX FIFO mode, applies to non-isochronous IN endpoints only. In dedicated FIFO mode, applies only to Control IN endpoints. In Scatter/Gather DMA mode, the TimeOUT interrupt is not asserted. Indicates that the core has detected a timeout condition on the USB For the last IN token on this endpoint. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
2	ahberr	AHB Error (AHBErr) Applies to IN and OUT endpoints. This is generated only in Internal DMA mode when there is an AHB error during an AHB read/write. The application can read the corresponding endpoint DMA address register to get the error address. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
1	epdisbld	Endpoint Disabled Interrupt (EPDisbld) Applies to IN and OUT endpoints. This bit indicates that the endpoint is disabled per the application's request. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
0	xfercompl	Transfer Completed Interrupt (XferCompl) Applies to IN and OUT endpoints. When Scatter/Gather DMA mode is enabled - For IN endpoint this field indicates that the requested data from the descriptor is moved from external system memory to internal FIFO. - For OUT endpoint this field indicates that the requested data from the internal FIFO is moved to external system memory. This interrupt is generated only when the corresponding endpoint descriptor is closed, and the IOC bit For the corresponding descriptor is Set. When Scatter/Gather DMA mode is disabled, this field indicates that the programmed transfer is complete on the AHB as well as on the USB, For this endpoint. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

dieptsiz2

Device IN Endpoint 2 Transfer Size Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00950
i_usbotg_1_devgrp	0xFFB40800	0xFFB40950

Offset: 0x150

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	mc RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
xfersize RW 0x0															

dieptsiz2 Fields

Bit	Name	Description	Access	Reset
30:29	mc	Applies to IN endpoints only. For periodic IN endpoints, this field indicates the number of packets that must be transmitted per microframe on the USB. The core uses this field to calculate the data PID for isochronous IN endpoints. 2'b01: 1 packet 2'b10: 2 packets 2'b11: 3 packets For non-periodic IN endpoints, this field is valid only in Internal DMA mode. It specifies the number of packets the core must fetchfor an IN endpoint before it switches to the endpoint pointed to by the Next Endpoint field of the Device Endpoint-n Control register (DIEPCTLn.NextEp) **Value** — **Description** 0x1 — PACKETONE 0x2 — PACKETTWO 0x3 — PACKETTHREE	RW	0x0

Bit	Name	Description	Access	Reset
28:19	pktcnt	Packet Count (PktCnt) Indicates the total number of USB packets that constitute the Transfer Size amount of data For endpoint 0. This field is decremented every time a packet (maximum size or short packet) is read from the TxFIFO.	RW	0x0
18:0	xfersize	Transfer Size (XferSize) Indicates the transfer size in bytes For endpoint 0. The core interrupts the application only after it has exhausted the transfer size amount of data. The transfer size can be Set to the maximum packet size of the endpoint, to be interrupted at the end of each packet. The core decrements this field every time a packet from the external memory is written to the TxFIFO.	RW	0x0

diepdma2

Device IN Endpoint 2 DMA Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00954
i_usbotg_1_devgrp	0xFFB40800	0xFFB40954

Offset: 0x154

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepdma2 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
diepdma2 RW 0x0															

diepdma2 Fields

Bit	Name	Description	Access	Reset
31:0	diepdma2	Holds the start address of the external memory for storing or fetching endpoint data. Note: For control endpoints, this field stores control OUT data packets as well as SETUP transaction data packets. When more than three SETUP packets are received back-to-back, the SETUP data packet in the memory is overwritten. This register is incremented on every AHB transaction. The application can give only a DWORD-aligned address. When Scatter/Gather DMA mode is not enabled, the application programs the start address value in this field. When Scatter/Gather DMA mode is enabled, this field indicates the base pointer for the descriptor list.	RW	0x0

dtxfsts2

Device IN Endpoint Transmit FIFO Status Register 2

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00958
i_usbotg_1_devgrp	0xFFB40800	0xFFB40958

Offset: 0x158

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ineptxfspcavail RO 0x2000															

dtxfsts2 Fields

Bit	Name	Description	Access	Reset
15:0	ineptxfspcavail	IN Endpoint TxFIFO Space Avail (INEPTxFSpcAvail) Indicates the amount of free space available in the Endpoint TxFIFO. Values are in terms of 32-bit words. 16'h0: Endpoint TxFIFO is full 16'h1: 1 word available 16'h2: 2 words available 16'hn: n words available (where 0 n 32,768) 16'h8000: 32,768 words available Others: Reserved	RO	0x2000

diepdmab2

Device IN Endpoint 2 Buffer Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB0095C
i_usbotg_1_devgrp	0xFFB40800	0xFFB4095C

Offset: 0x15C

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepdmab2 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
diepdmab2 RO 0x0															

diepdmab2 Fields

Bit	Name	Description	Access	Reset
31:0	diepdmab2	Holds the current buffer address.This register is updated as and when the data transfer for the corresponding end point is in progress. This register is present only in Scatter/Gather DMA mode. Otherwise this field is reserved.	RO	0x0

diepctl3

Device Control IN Endpoint 3 Control Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00960
i_usbotg_1_devgrp	0xFFB40800	0xFFB40960

Offset: 0x160

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	txfnum RW 0x0				stall RW 0x0	Reserved	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
usbactep RW 0x0	Reserved				mps RW 0x0										

diepctl3 Fields

Bit	Name	Description	Access	Reset
31	epena	Endpoint Enable (EPEna) Applies to IN and OUT endpoints. When Scatter/Gather DMA mode is enabled, For IN endpoints this bit indicates that the descriptor structure and data buffer with data ready to transmit is setup. For OUT endpoint it indicates that the descriptor structure and data buffer to receive data is setup. When Scatter/Gather DMA mode is enabledsuch as for buffer-pointer based DMA mode: - For IN endpoints, this bit indicates that data is ready to be transmitted on the endpoint. - For OUT endpoints, this bit indicates that the application has allocated the memory to start receiving data from the USB. - The core clears this bit before setting any of the following interrupts on this endpoint: SETUP Phase Done Endpoint Disabled Transfer Completed Note: For control endpoints in DMA mode, this bit must be set to be able to transfer SETUP data packets in memory. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
30	epdis	Endpoint Disable (EPDis) Applies to IN and OUT endpoints. The application sets this bit to stop transmitting/receiving data on an endpoint, even before the transfer for that endpoint is complete. The application must wait for the Endpoint Disabled interrupt before treating the endpoint as disabled. The core clears this bit before setting the Endpoint Disabled interrupt. The application must set this bit only if Endpoint Enable is already set for this endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
29	setd1pid	Set DATA1 PID (SetD1PID) Applies to interrupt/bulk IN and OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA1. This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. Set Odd (micro)frame (SetOddFr) Applies to isochronous IN and OUT endpoints only. Writing to this field sets the Even/Odd (micro)frame (EO_FrNum) field to odd (micro)frame. This field is not applicable for Scatter/Gather DMA mode. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	WO	0x0

Bit	Name	Description	Access	Reset
28	setd0pid	Set DATA0 PID (SetD0PID) Applies to interrupt/bulk IN and OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA0. This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. 1'b0 WO In non-Scatter/Gather DMA mode: Set Even (micro)frame (SetEvenFr) Applies to isochronous IN and OUT endpoints only. Writing to this field sets the Even/Odd (micro)frame (EO_FrNum) field to even (micro) frame. When Scatter/Gather DMA mode is enabled, this field is reserved. The frame number in which to send data is in the transmit descriptor structure. The frame in which to receive data is updated in receive descriptor structure. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	WO	0x0
27	snak	Set NAK (SNAK) A write to this bit sets the NAK bit For the endpoint. Using this bit, the application can control the transmission of NAK handshakes on an endpoint. The core can also Set this bit For an endpoint after a SETUP packet is received on that endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	WO	0x0

Bit	Name	Description	Access	Reset
26	cnak	Clear NAK (CNAK) A write to this bit clears the NAK bit For the endpoint. Value: 0x0 — Description: INACTIVE Value: 0x1 — Description: ACTIVE	WO	0x0
25:22	txfnum	TxFIFO Number (TxFNum) Shared FIFO Operation non-periodic endpoints must set this bit to zero. Periodic endpoints must map this to the corresponding Periodic TxFIFO number. 4'h0: Non-Periodic TxFIFO Others: Specified Periodic TxFIFO.number Note: An interrupt IN endpoint can be configured as a non-periodic endpoint for applications such as mass storage. The core treats an IN endpoint as a non-periodic endpoint if the TxFNum field is set to 0. Otherwise, a separate periodic FIFO must be allocated for an interrupt IN endpoint, and the number of this FIFO must be programmed into the TxFNum field. Configuring an interrupt IN endpoint as a non-periodic endpoint saves the extra periodic FIFO area. Dedicated FIFO Operationthese bits specify the FIFO number associated with this endpoint. Each active IN endpoint must be programmed to a separate FIFO number. This field is valid only for IN endpoints.	RW	0x0

Bit	Name	Description	Access	Reset
21	stall	STALL Handshake (Stall) Applies to non-control, non-isochronous IN and OUT endpoints only. The application sets this bit to stall all tokens from the USB host to this endpoint. If a NAK bit, Global Non-periodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority. Only the application can clear this bit, never the core. 1'b0 R_W Applies to control endpoints only. The application can only set this bit, and the core clears it, when a SETUP token is received for this endpoint. If a NAK bit, Global Non-periodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority. Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake. **Value: Description** 0x0: INACTIVE 0x1: ACTIVE	RW	0x0
19:18	eptype	Endpoint Type (EPType) This is the transfer type supported by this logical endpoint. 2'b00: Control 2'b01: Isochronous 2'b10: Bulk 2'b11: Interrupt **Value: Description** 0x0: CONTROL 0x1: ISOCHRONOUS 0x2: BULK 0x3: INTERRUP	RW	0x0

Bit	Name	Description	Access	Reset
17	naksts	NAK Status (NAKSts) Indicates the following: 1'b0: The core is transmitting non-NAK handshakes based on the FIFO status. 1'b1: The core is transmitting NAK handshakes on this endpoint. When either the application or the core sets this bit: The core stops receiving any data on an OUT endpoint, even if there is space in the RxFIFO to accommodate the incoming packet. For non-isochronous IN endpoints: The core stops transmitting any data on an IN endpoint, even if there data is available in the TxFIFO. For isochronous IN endpoints: The core sends out a zero-length data packet, even if there data is available in the TxFIFO. Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake. **Value** — **Description** 0x0 — NONNAK 0x1 — NAK	RO	0x0

Bit	Name	Description	Access	Reset
16	dpid	Endpoint Data PID (DPID) Applies to interrupt/bulk IN and OUT endpoints only. Contains the PID of the packet to be received or transmitted on this endpoint. The application must program the PID of the first packet to be received or transmitted on this endpoint, after the endpoint is activated. The applications use the SetD1PID and SetD0PID fields of this register to program either DATA0 or DATA1 PID. 1'b0: DATA0 1'b1: DATA1 This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. 1'b0 RO Even/Odd (Micro)Frame (EO_FrNum) In non-Scatter/Gather DMA mode: Applies to isochronous IN and OUT endpoints only. Indicates the (micro)frame number in which the core transmits/ receives isochronous data for this endpoint. The application must program the even/odd (micro) frame number in which it intends to transmit/receive isochronous data for this endpoint using the SetEvnFr and SetOddFr fields in this register. 1'b0: Even (micro)frame 1'b1: Odd (micro)frame When Scatter/Gather DMA mode is enabled, this field is reserved. The frame number in which to send data is provided in the transmit descriptor structure. The frame in which data is received is updated in receive descriptor structure. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RO	0x0

Bit	Name	Description	Access	Reset
15	usbactep	USB Active Endpoint (USBActEP) Indicates whether this endpoint is active in the current configuration and interface. The core clears this bit for all endpoints (other than EP 0) after detecting a USB reset. After receiving the SetConfiguration and SetInterface commands, the applica-tion must program endpoint registers accordingly and set this bit. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
10:0	mps	Maximum Packet Size (MPS) The application must program this field with the maximum packet size for the current logical endpoint. This value is in bytes.	RW	0x0

diepint3

Device IN Endpoint 3 Interrupt Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00968
i_usbotg_1_devgrp	0xFFB40800	0xFFB40968

Offset: 0x168

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	txfifoundrn RW 0x0	txfempp RO 0x1	inepnakeff RW 0x0	intknepmis RW 0x0	intkntxfempp RW 0x0	timeout RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0

diepint3 Fields

Bit	Name	Description	Access	Reset
14	nyetintrpt	NYET Interrupt (NYETIntrpt) The core generates this interrupt when a NYET response is transmitted for a non isochronous OUT endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
13	nakintrpt	NAK Interrupt (NAKInterrupt) The core generates this interrupt when a NAK is transmitted or received by the device. In case of isochronous IN endpoints the interrupt gets generated when a zero length packet is transmitted due to un-availability of data in the TXFifo. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
12	bbleerr	NAK Interrupt (BbleErr) The core generates this interrupt when babble is received for the endpoint. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
11	pktdrpsts	Packet Drop Status (PktDrpSts) This bit indicates to the application that an ISOC OUT packet has been dropped. This bit does not have an associated mask bit and does not generate an interrupt. Dependency: This bit is valid in non Scatter/Gather DMA mode when periodic transfer interrupt feature is selected. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
9	bnaintr	BNA (Buffer Not Available) Interrupt (BNAIntr) This bit is valid only when Scatter/Gather DMA mode is enabled. The core generates this interrupt when the descriptor accessed is not ready For the Core to process, such as Host busy or DMA done **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
8	txfifoundrn	Fifo Underrun (TxfifoUndrn) Applies to IN endpoints Only This bit is valid only If thresholding is enabled. The core generates this interrupt when it detects a transmit FIFO underrun condition For this endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
7	txfemp	Transmit FIFO Empty (TxFEmp) This bit is valid only For IN Endpoints This interrupt is asserted when the TxFIFO For this endpoint is either half or completely empty. The half or completely empty status is determined by the TxFIFO Empty Level bit in the Core AHB Configuration register (GAHBCFG.NPTxFEmpLvl)). **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RO	0x1

Bit	Name	Description	Access	Reset
6	inepnakeff	IN Endpoint NAK Effective (INEPNakEff) Applies to periodic IN endpoints only. This bit can be cleared when the application clears the IN endpoint NAK by writing to DIEPCTLn.CNAK. This interrupt indicates that the core has sampled the NAK bit Set (either by the application or by the core). The interrupt indicates that the IN endpoint NAK bit Set by the application has taken effect in the core. This interrupt does not guarantee that a NAK handshake is sent on the USB. A STALL bit takes priority over a NAK bit. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
5	intknepmis	IN Token Received with EP Mismatch (INTknEPMis) Applies to non-periodic IN endpoints only. Indicates that the data in the top of the non-periodic TxFIFO belongs to an endpoint other than the one For which the IN token was received. This interrupt is asserted on the endpoint For which the IN token was received. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
4	intkntxfemp	IN Token Received When TxFIFO is Empty (INTknTXFEmp) Applies to non-periodic IN endpoints only. Indicates that an IN token was received when the associated TxFIFO (periodic/non-periodic) was empty. This interrupt is asserted on the endpoint For which the IN token was received. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
3	timeout	Timeout Condition (TimeOUT) In shared TX FIFO mode, applies to non-isochronous IN endpoints only. In dedicated FIFO mode, applies only to Control IN endpoints. In Scatter/Gather DMA mode, the TimeOUT interrupt is not asserted. Indicates that the core has detected a timeout condition on the USB For the last IN token on this endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
2	ahberr	AHB Error (AHBErr) Applies to IN and OUT endpoints. This is generated only in Internal DMA mode when there is an AHB error during an AHB read/write. The application can read the corresponding endpoint DMA address register to get the error address. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
1	epdisbld	Endpoint Disabled Interrupt (EPDisbld) Applies to IN and OUT endpoints. This bit indicates that the endpoint is disabled per the application's request. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
0	xfercompl	Transfer Completed Interrupt (XferCompl) Applies to IN and OUT endpoints. When Scatter/Gather DMA mode is enabled - For IN endpoint this field indicates that the requested data from the descriptor is moved from external system memory to internal FIFO. - For OUT endpoint this field indicates that the requested data from the internal FIFO is moved to external system memory. This interrupt is generated only when the corresponding endpoint descriptor is closed, and the IOC bit For the corresponding descriptor is Set. When Scatter/Gather DMA mode is disabled, this field indicates that the programmed transfer is complete on the AHB as well as on the USB, For this endpoint. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

dieptsiz3

Device IN Endpoint 3 Transfer Size Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00970
i_usbotg_1_devgrp	0xFFB40800	0xFFB40970

Offset: 0x170

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	mc RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
xfersize RW 0x0															

dieptsiz3 Fields

Bit	Name	Description	Access	Reset
30:29	mc	Applies to IN endpoints only. For periodic IN endpoints, this field indicates the number of packets that must be transmitted per microframe on the USB. The core uses this field to calculate the data PID for isochronous IN endpoints. 2'b01: 1 packet 2'b10: 2 packets 2'b11: 3 packets For non-periodic IN endpoints, this field is valid only in Internal DMA mode. It specifies the number of packets the core must fetchfor an IN endpoint before it switches to the endpoint pointed to by the Next Endpoint field of the Device Endpoint-n Control register (DIEPCTLn.NextEp) **Value** — **Description** 0x1 — PACKETONE 0x2 — PACKETTWO 0x3 — PACKETTHREE	RW	0x0

Bit	Name	Description	Access	Reset
28:19	pktcnt	Packet Count (PktCnt) Indicates the total number of USB packets that constitute the Transfer Size amount of data For endpoint 0. This field is decremented every time a packet (maximum size or short packet) is read from the TxFIFO.	RW	0x0
18:0	xfersize	Transfer Size (XferSize) Indicates the transfer size in bytes For endpoint 0. The core interrupts the application only after it has exhausted the transfer size amount of data. The transfer size can be Set to the maximum packet size of the endpoint, to be interrupted at the end of each packet. The core decrements this field every time a packet from the external memory is written to the TxFIFO.	RW	0x0

diepdma3

Device IN Endpoint 3 DMA Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00974
i_usbotg_1_devgrp	0xFFB40800	0xFFB40974

Offset: 0x174

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepdma3 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
diepdma3 RW 0x0															

diepdma3 Fields

Bit	Name	Description	Access	Reset
31:0	diepdma3	Holds the start address of the external memory for storing or fetching endpoint data. Note: For control endpoints, this field stores control OUT data packets as well as SETUP transaction data packets. When more than three SETUP packets are received back-to-back, the SETUP data packet in the memory is overwritten. This register is incremented on every AHB transaction. The application can give only a DWORD-aligned address. When Scatter/Gather DMA mode is not enabled, the application programs the start address value in this field. When Scatter/Gather DMA mode is enabled, this field indicates the base pointer for the descriptor list.	RW	0x0

dtxfsts3

Device IN Endpoint Transmit FIFO Status Register 3

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00978
i_usbotg_1_devgrp	0xFFB40800	0xFFB40978

Offset: 0x178

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ineptxfspcavail RO 0x2000															

dtxfsts3 Fields

Bit	Name	Description	Access	Reset
15:0	ineptxfspcavail	IN Endpoint TxFIFO Space Avail (INEPTxFSpcAvail) Indicates the amount of free space available in the Endpoint TxFIFO. Values are in terms of 32-bit words. 16'h0: Endpoint TxFIFO is full 16'h1: 1 word available 16'h2: 2 words available 16'hn: n words available (where 0 n 32,768) 16'h8000: 32,768 words available Others: Reserved	RO	0x2000

diepdmab3

Device IN Endpoint 3 Buffer Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB0097C
i_usbotg_1_devgrp	0xFFB40800	0xFFB4097C

Offset: 0x17C

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepdmab3 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
diepdmab3 RO 0x0															

diepdmab3 Fields

Bit	Name	Description	Access	Reset
31:0	diepdmab3	Holds the current buffer address.This register is updated as and when the data transfer for the corresponding end point is in progress. This register is present only in Scatter/Gather DMA mode. Otherwise this field is reserved.	RO	0x0

diepctl4

Device Control IN Endpoint 4 Control Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00980
i_usbotg_1_devgrp	0xFFB40800	0xFFB40980

Offset: 0x180

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	txfnum RW 0x0				stall RW 0x0	Reserved	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
usbactep RW 0x0	Reserved				mps RW 0x0										

diepctl4 Fields

Bit	Name	Description	Access	Reset
31	epena	Endpoint Enable (EPEna) Applies to IN and OUT endpoints. When Scatter/Gather DMA mode is enabled, For IN endpoints this bit indicates that the descriptor structure and data buffer with data ready to transmit is setup. For OUT endpoint it indicates that the descriptor structure and data buffer to receive data is setup. When Scatter/Gather DMA mode is enabledsuch as for buffer-pointer based DMA mode: - For IN endpoints, this bit indicates that data is ready to be transmitted on the endpoint. - For OUT endpoints, this bit indicates that the application has allocated the memory to start receiving data from the USB. - The core clears this bit before setting any of the following interrupts on this endpoint: SETUP Phase Done Endpoint Disabled Transfer Completed Note: For control endpoints in DMA mode, this bit must be set to be able to transfer SETUP data packets in memory. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
30	epdis	Endpoint Disable (EPDis) Applies to IN and OUT endpoints. The application sets this bit to stop transmitting/receiving data on an endpoint, even before the transfer for that endpoint is complete. The application must wait for the Endpoint Disabled interrupt before treating the endpoint as disabled. The core clears this bit before setting the Endpoint Disabled interrupt. The application must set this bit only if Endpoint Enable is already set for this endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
29	setd1pid	Set DATA1 PID (SetD1PID) Applies to interrupt/bulk IN and OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA1. This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. Set Odd (micro)frame (SetOddFr) Applies to isochronous IN and OUT endpoints only. Writing to this field sets the Even/Odd (micro)frame (EO_FrNum) field to odd (micro)frame. This field is not applicable for Scatter/Gather DMA mode. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	WO	0x0

Bit	Name	Description	Access	Reset
28	setd0pid	Set DATA0 PID (SetD0PID) Applies to interrupt/bulk IN and OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA0. This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. 1'b0 WO In non-Scatter/Gather DMA mode: Set Even (micro)frame (SetEvenFr) Applies to isochronous IN and OUT endpoints only. Writing to this field sets the Even/Odd (micro)frame (EO_FrNum) field to even (micro) frame. When Scatter/Gather DMA mode is enabled, this field is reserved. The frame number in which to send data is in the transmit descriptor structure. The frame in which to receive data is updated in receive descriptor structure. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	WO	0x0
27	snak	Set NAK (SNAK) A write to this bit sets the NAK bit For the endpoint. Using this bit, the application can control the transmission of NAK handshakes on an endpoint. The core can also Set this bit For an endpoint after a SETUP packet is received on that endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	WO	0x0

Bit	Name	Description	Access	Reset
26	cnak	Clear NAK (CNAK) A write to this bit clears the NAK bit For the endpoint. Value: 0x0 — Description: INACTIVE Value: 0x1 — Description: ACTIVE	WO	0x0
25:22	txfnum	TxFIFO Number (TxFNum) Shared FIFO Operation non-periodic endpoints must set this bit to zero. Periodic endpoints must map this to the corresponding Periodic TxFIFO number. 4'h0: Non-Periodic TxFIFO Others: Specified Periodic TxFIFO.number Note: An interrupt IN endpoint can be configured as a non-periodic endpoint for applications such as mass storage. The core treats an IN endpoint as a non-periodic endpoint if the TxFNum field is set to 0. Otherwise, a separate periodic FIFO must be allocated for an interrupt IN endpoint, and the number of this FIFO must be programmed into the TxFNum field. Configuring an interrupt IN endpoint as a non-periodic endpoint saves the extra periodic FIFO area. Dedicated FIFO Operationthese bits specify the FIFO number associated with this endpoint. Each active IN endpoint must be programmed to a separate FIFO number. This field is valid only for IN endpoints.	RW	0x0

Bit	Name	Description	Access	Reset
21	stall	STALL Handshake (Stall) Applies to non-control, non-isochronous IN and OUT endpoints only. The application sets this bit to stall all tokens from the USB host to this endpoint. If a NAK bit, Global Non-periodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority. Only the application can clear this bit, never the core. 1'b0 R_W Applies to control endpoints only. The application can only set this bit, and the core clears it, when a SETUP token is received for this endpoint. If a NAK bit, Global Non-periodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority. Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
19:18	eptype	Endpoint Type (EPType) This is the transfer type supported by this logical endpoint. 2'b00: Control 2'b01: Isochronous 2'b10: Bulk 2'b11: Interrupt **Value** **Description** 0x0 CONTROL 0x1 ISOCHRONOUS 0x2 BULK 0x3 INTERRUP	RW	0x0

Bit	Name	Description	Access	Reset
17	naksts	NAK Status (NAKSts) Indicates the following: 1'b0: The core is transmitting non-NAK handshakes based on the FIFO status. 1'b1: The core is transmitting NAK handshakes on this endpoint. When either the application or the core sets this bit: The core stops receiving any data on an OUT endpoint, even if there is space in the RxFIFO to accommodate the incoming packet. For non-isochronous IN endpoints: The core stops transmitting any data on an IN endpoint, even if there data is available in the TxFIFO. For isochronous IN endpoints: The core sends out a zero-length data packet, even if there data is available in the TxFIFO. Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake. **Value** — **Description** 0x0 — NONNAK 0x1 — NAK	RO	0x0

Bit	Name	Description	Access	Reset
16	dpid	Endpoint Data PID (DPID) Applies to interrupt/bulk IN and OUT endpoints only. Contains the PID of the packet to be received or transmitted on this endpoint. The application must program the PID of the first packet to be received or transmitted on this endpoint, after the endpoint is activated. The applications use the SetD1PID and SetD0PID fields of this register to program either DATA0 or DATA1 PID. 1'b0: DATA0 1'b1: DATA1 This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. 1'b0 RO Even/Odd (Micro)Frame (EO_FrNum) In non-Scatter/Gather DMA mode: Applies to isochronous IN and OUT endpoints only. Indicates the (micro)frame number in which the core transmits/ receives isochronous data for this endpoint. The application must program the even/odd (micro) frame number in which it intends to transmit/receive isochronous data for this endpoint using the SetEvnFr and SetOddFr fields in this register. 1'b0: Even (micro)frame 1'b1: Odd (micro)frame When Scatter/Gather DMA mode is enabled, this field is reserved. The frame number in which to send data is provided in the transmit descriptor structure. The frame in which data is received is updated in receive descriptor structure. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RO	0x0

Bit	Name	Description	Access	Reset
15	usbactep	USB Active Endpoint (USBActEP) Indicates whether this endpoint is active in the current configuration and interface. The core clears this bit for all endpoints (other than EP 0) after detecting a USB reset. After receiving the SetConfiguration and SetInterface commands, the applica-tion must program endpoint registers accordingly and set this bit. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
10:0	mps	Maximum Packet Size (MPS) The application must program this field with the maximum packet size for the current logical endpoint. This value is in bytes.	RW	0x0

diepint4

Device IN Endpoint 4 Interrupt Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00988
i_usbotg_1_devgrp	0xFFB40800	0xFFB40988

Offset: 0x188

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	txfifoundrn RW 0x0	txfempp RO 0x1	inepnakeff RW 0x0	intknepmis RW 0x0	intkntxfempp RW 0x0	timeout RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0

diepint4 Fields

Bit	Name	Description	Access	Reset
14	nyetintrpt	NYET Interrupt (NYETIntrpt) The core generates this interrupt when a NYET response is transmitted for a non isochronous OUT endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
13	nakintrpt	NAK Interrupt (NAKInterrupt) The core generates this interrupt when a NAK is transmitted or received by the device. In case of isochronous IN endpoints the interrupt gets generated when a zero length packet is transmitted due to un-availability of data in the TXFifo. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
12	bbleerr	NAK Interrupt (BbleErr) The core generates this interrupt when babble is received for the endpoint. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
11	pktdrpsts	Packet Drop Status (PktDrpSts) This bit indicates to the application that an ISOC OUT packet has been dropped. This bit does not have an associated mask bit and does not generate an interrupt. Dependency: This bit is valid in non Scatter/Gather DMA mode when periodic transfer interrupt feature is selected. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
9	bnaintr	BNA (Buffer Not Available) Interrupt (BNAIntr) This bit is valid only when Scatter/Gather DMA mode is enabled. The core generates this interrupt when the descriptor accessed is not ready For the Core to process, such as Host busy or DMA done **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
8	txfifoundrn	Fifo Underrun (TxfifoUndrn) Applies to IN endpoints Only This bit is valid only If thresholding is enabled. The core generates this interrupt when it detects a transmit FIFO underrun condition For this endpoint. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
7	txfemp	Transmit FIFO Empty (TxFEmp) This bit is valid only For IN Endpoints This interrupt is asserted when the TxFIFO For this endpoint is either half or completely empty. The half or completely empty status is determined by the TxFIFO Empty Level bit in the Core AHB Configuration register (GAHBCFG.NPTxFEmpLvl)). **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x1

Bit	Name	Description	Access	Reset
6	inepnakeff	IN Endpoint NAK Effective (INEPNakEff) Applies to periodic IN endpoints only. This bit can be cleared when the application clears the IN endpoint NAK by writing to DIEPCTLn.CNAK. This interrupt indicates that the core has sampled the NAK bit Set (either by the application or by the core). The interrupt indicates that the IN endpoint NAK bit Set by the application has taken effect in the core. This interrupt does not guarantee that a NAK handshake is sent on the USB. A STALL bit takes priority over a NAK bit. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
5	intknepmis	IN Token Received with EP Mismatch (INTknEPMis) Applies to non-periodic IN endpoints only. Indicates that the data in the top of the non-periodic TxFIFO belongs to an endpoint other than the one For which the IN token was received. This interrupt is asserted on the endpoint For which the IN token was received. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
4	intkntxfemp	IN Token Received When TxFIFO is Empty (INTknTXFEmp) Applies to non-periodic IN endpoints only. Indicates that an IN token was received when the associated TxFIFO (periodic/non-periodic) was empty. This interrupt is asserted on the endpoint For which the IN token was received. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
3	timeout	Timeout Condition (TimeOUT) In shared TX FIFO mode, applies to non-isochronous IN endpoints only. In dedicated FIFO mode, applies only to Control IN endpoints. In Scatter/Gather DMA mode, the TimeOUT interrupt is not asserted. Indicates that the core has detected a timeout condition on the USB For the last IN token on this endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
2	ahberr	AHB Error (AHBErr) Applies to IN and OUT endpoints. This is generated only in Internal DMA mode when there is an AHB error during an AHB read/write. The application can read the corresponding endpoint DMA address register to get the error address. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
1	epdisbld	Endpoint Disabled Interrupt (EPDisbld) Applies to IN and OUT endpoints. This bit indicates that the endpoint is disabled per the application's request. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
0	xfercompl	Transfer Completed Interrupt (XferCompl) Applies to IN and OUT endpoints. When Scatter/Gather DMA mode is enabled - For IN endpoint this field indicates that the requested data from the descriptor is moved from external system memory to internal FIFO. - For OUT endpoint this field indicates that the requested data from the internal FIFO is moved to external system memory. This interrupt is generated only when the corresponding endpoint descriptor is closed, and the IOC bit For the corresponding descriptor is Set. When Scatter/Gather DMA mode is disabled, this field indicates that the programmed transfer is complete on the AHB as well as on the USB, For this endpoint. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

dieptsiz4

Device IN Endpoint 4 Transfer Size Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00990
i_usbotg_1_devgrp	0xFFB40800	0xFFB40990

Offset: 0x190

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	mc RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
xfersize RW 0x0															

dieptsiz4 Fields

Bit	Name	Description	Access	Reset
30:29	mc	Applies to IN endpoints only. For periodic IN endpoints, this field indicates the number of packets that must be transmitted per microframe on the USB. The core uses this field to calculate the data PID for isochronous IN endpoints. 2'b01: 1 packet 2'b10: 2 packets 2'b11: 3 packets For non-periodic IN endpoints, this field is valid only in Internal DMA mode. It specifies the number of packets the core must fetchfor an IN endpoint before it switches to the endpoint pointed to by the Next Endpoint field of the Device Endpoint-n Control register (DIEPCTLn.NextEp) **Value** — **Description** 0x1 — PACKETONE 0x2 — PACKETTWO 0x3 — PACKETTHREE	RW	0x0

Bit	Name	Description	Access	Reset
28:19	pktcnt	Packet Count (PktCnt) Indicates the total number of USB packets that constitute the Transfer Size amount of data For endpoint 0. This field is decremented every time a packet (maximum size or short packet) is read from the TxFIFO.	RW	0x0
18:0	xfersize	Transfer Size (XferSize) Indicates the transfer size in bytes For endpoint 0. The core interrupts the application only after it has exhausted the transfer size amount of data. The transfer size can be Set to the maximum packet size of the endpoint, to be interrupted at the end of each packet. The core decrements this field every time a packet from the external memory is written to the TxFIFO.	RW	0x0

diepdma4

Device IN Endpoint 4 DMA Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00994
i_usbotg_1_devgrp	0xFFB40800	0xFFB40994

Offset: 0x194

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepdma4 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
diepdma4 RW 0x0															

diepdma4 Fields

Bit	Name	Description	Access	Reset
31:0	diepdma4	Holds the start address of the external memory for storing or fetching endpoint data. Note: For control endpoints, this field stores control OUT data packets as well as SETUP transaction data packets. When more than three SETUP packets are received back-to-back, the SETUP data packet in the memory is overwritten. This register is incremented on every AHB transaction. The application can give only a DWORD-aligned address. When Scatter/Gather DMA mode is not enabled, the application programs the start address value in this field. When Scatter/Gather DMA mode is enabled, this field indicates the base pointer for the descriptor list.	RW	0x0

dtxfsts4

Device IN Endpoint Transmit FIFO Status Register 4

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00998
i_usbotg_1_devgrp	0xFFB40800	0xFFB40998

Offset: 0x198

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ineptxfspcavail RO 0x2000															

dtxfsts4 Fields

Bit	Name	Description	Access	Reset
15:0	ineptxfspcavail	IN Endpoint TxFIFO Space Avail (INEPTxFSpcAvail) Indicates the amount of free space available in the Endpoint TxFIFO. Values are in terms of 32-bit words. 16'h0: Endpoint TxFIFO is full 16'h1: 1 word available 16'h2: 2 words available 16'hn: n words available (where 0 n 32,768) 16'h8000: 32,768 words available Others: Reserved	RO	0x2000

diepdmab4

Device IN Endpoint 4 Buffer Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB0099C
i_usbotg_1_devgrp	0xFFB40800	0xFFB4099C

Offset: 0x19C

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepdmab4 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
diepdmab4 RO 0x0															

diepdmab4 Fields

Bit	Name	Description	Access	Reset
31:0	diepdmab4	Holds the current buffer address.This register is updated as and when the data transfer for the corresponding end point is in progress. This register is present only in Scatter/Gather DMA mode. Otherwise this field is reserved.	RO	0x0

diepctl5

Device Control IN Endpoint 5 Control Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB009A0
i_usbotg_1_devgrp	0xFFB40800	0xFFB409A0

Offset: 0x1A0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	txfnum RW 0x0				stall RW 0x0	Reserved	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
usbactep RW 0x0	Reserved				mps RW 0x0										

diepctl5 Fields

Bit	Name	Description	Access	Reset
31	epena	Endpoint Enable (EPEna) Applies to IN and OUT endpoints. When Scatter/Gather DMA mode is enabled, For IN endpoints this bit indicates that the descriptor structure and data buffer with data ready to transmit is setup. For OUT endpoint it indicates that the descriptor structure and data buffer to receive data is setup. When Scatter/Gather DMA mode is enabledsuch as for buffer-pointer based DMA mode: - For IN endpoints, this bit indicates that data is ready to be transmitted on the endpoint. - For OUT endpoints, this bit indicates that the application has allocated the memory to start receiving data from the USB. - The core clears this bit before setting any of the following interrupts on this endpoint: SETUP Phase Done Endpoint Disabled Transfer Completed Note: For control endpoints in DMA mode, this bit must be set to be able to transfer SETUP data packets in memory. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
30	epdis	Endpoint Disable (EPDis) Applies to IN and OUT endpoints. The application sets this bit to stop transmitting/receiving data on an endpoint, even before the transfer for that endpoint is complete. The application must wait for the Endpoint Disabled interrupt before treating the endpoint as disabled. The core clears this bit before setting the Endpoint Disabled interrupt. The application must set this bit only if Endpoint Enable is already set for this endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
29	setd1pid	Set DATA1 PID (SetD1PID) Applies to interrupt/bulk IN and OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA1. This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. Set Odd (micro)frame (SetOddFr) Applies to isochronous IN and OUT endpoints only. Writing to this field sets the Even/Odd (micro)frame (EO_FrNum) field to odd (micro)frame. This field is not applicable for Scatter/Gather DMA mode. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	WO	0x0

Bit	Name	Description	Access	Reset
28	setd0pid	Set DATA0 PID (SetD0PID) Applies to interrupt/bulk IN and OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA0. This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. 1'b0 WO In non-Scatter/Gather DMA mode: Set Even (micro)frame (SetEvenFr) Applies to isochronous IN and OUT endpoints only. Writing to this field sets the Even/Odd (micro)frame (EO_FrNum) field to even (micro) frame. When Scatter/Gather DMA mode is enabled, this field is reserved. The frame number in which to send data is in the transmit descriptor structure. The frame in which to receive data is updated in receive descriptor structure. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	WO	0x0
27	snak	Set NAK (SNAK) A write to this bit sets the NAK bit For the endpoint. Using this bit, the application can control the transmission of NAK handshakes on an endpoint. The core can also Set this bit For an endpoint after a SETUP packet is received on that endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	WO	0x0

Bit	Name	Description	Access	Reset
26	cnak	Clear NAK (CNAK) A write to this bit clears the NAK bit For the endpoint. Value: 0x0 — INACTIVE Value: 0x1 — ACTIVE	WO	0x0
25:22	txfnum	TxFIFO Number (TxFNum) Shared FIFO Operation non-periodic endpoints must set this bit to zero. Periodic endpoints must map this to the corresponding Periodic TxFIFO number. 4'h0: Non-Periodic TxFIFO Others: Specified Periodic TxFIFO.number Note: An interrupt IN endpoint can be configured as a non-periodic endpoint for applications such as mass storage. The core treats an IN endpoint as a non-periodic endpoint if the TxFNum field is set to 0. Otherwise, a separate periodic FIFO must be allocated for an interrupt IN endpoint, and the number of this FIFO must be programmed into the TxFNum field. Configuring an interrupt IN endpoint as a non-periodic endpoint saves the extra periodic FIFO area. Dedicated FIFO Operationthese bits specify the FIFO number associated with this endpoint. Each active IN endpoint must be programmed to a separate FIFO number. This field is valid only for IN endpoints.	RW	0x0

Bit	Name	Description	Access	Reset
21	stall	STALL Handshake (Stall) Applies to non-control, non-isochronous IN and OUT endpoints only. The application sets this bit to stall all tokens from the USB host to this endpoint. If a NAK bit, Global Non-periodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority. Only the application can clear this bit, never the core. 1'b0 R_W Applies to control endpoints only. The application can only set this bit, and the core clears it, when a SETUP token is received for this endpoint. If a NAK bit, Global Non-periodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority. Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake. **Value: Description** 0x0: INACTIVE 0x1: ACTIVE	RW	0x0
19:18	eptype	Endpoint Type (EPType) This is the transfer type supported by this logical endpoint. 2'b00: Control 2'b01: Isochronous 2'b10: Bulk 2'b11: Interrupt **Value: Description** 0x0: CONTROL 0x1: ISOCHRONOUS 0x2: BULK 0x3: INTERRUP	RW	0x0

Bit	Name	Description	Access	Reset
17	naksts	NAK Status (NAKSts) Indicates the following: 1'b0: The core is transmitting non-NAK handshakes based on the FIFO status. 1'b1: The core is transmitting NAK handshakes on this endpoint. When either the application or the core sets this bit: The core stops receiving any data on an OUT endpoint, even if there is space in the RxFIFO to accommodate the incoming packet. For non-isochronous IN endpoints: The core stops transmitting any data on an IN endpoint, even if there data is available in the TxFIFO. For isochronous IN endpoints: The core sends out a zero-length data packet, even if there data is available in the TxFIFO. Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake. **Value** — **Description** 0x0 — NONNAK 0x1 — NAK	RO	0x0

Bit	Name	Description	Access	Reset
16	dpid	Endpoint Data PID (DPID) Applies to interrupt/bulk IN and OUT endpoints only. Contains the PID of the packet to be received or transmitted on this endpoint. The application must program the PID of the first packet to be received or transmitted on this endpoint, after the endpoint is activated. The applications use the SetD1PID and SetD0PID fields of this register to program either DATA0 or DATA1 PID. 1'b0: DATA0 1'b1: DATA1 This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. 1'b0 RO Even/Odd (Micro)Frame (EO_FrNum) In non-Scatter/Gather DMA mode: Applies to isochronous IN and OUT endpoints only. Indicates the (micro)frame number in which the core transmits/ receives isochronous data for this endpoint. The application must program the even/odd (micro) frame number in which it intends to transmit/receive isochronous data for this endpoint using the SetEvnFr and SetOddFr fields in this register. 1'b0: Even (micro)frame 1'b1: Odd (micro)frame When Scatter/Gather DMA mode is enabled, this field is reserved. The frame number in which to send data is provided in the transmit descriptor structure. The frame in which data is received is updated in receive descriptor structure. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RO	0x0

Bit	Name	Description	Access	Reset
15	usbactep	USB Active Endpoint (USBActEP) Indicates whether this endpoint is active in the current configuration and interface. The core clears this bit for all endpoints (other than EP 0) after detecting a USB reset. After receiving the SetConfiguration and SetInterface commands, the applica-tion must program endpoint registers accordingly and set this bit. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
10:0	mps	Maximum Packet Size (MPS) The application must program this field with the maximum packet size for the current logical endpoint. This value is in bytes.	RW	0x0

diepint5

Device IN Endpoint 5 Interrupt Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB009A8
i_usbotg_1_devgrp	0xFFB40800	0xFFB409A8

Offset: 0x1A8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	txfifoundrn RW 0x0	txfempp RO 0x1	inepnakeff RW 0x0	intknepmis RW 0x0	intkntxfempp RW 0x0	timeout RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0

diepint5 Fields

Bit	Name	Description	Access	Reset
14	nyetintrpt	NYET Interrupt (NYETIntrpt) The core generates this interrupt when a NYET response is transmitted for a non isochronous OUT endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
13	nakintrpt	NAK Interrupt (NAKInterrupt) The core generates this interrupt when a NAK is transmitted or received by the device. In case of isochronous IN endpoints the interrupt gets generated when a zero length packet is transmitted due to un-availability of data in the TXFifo. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
12	bbleerr	NAK Interrupt (BbleErr) The core generates this interrupt when babble is received for the endpoint. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
11	pktdrpsts	Packet Drop Status (PktDrpSts) This bit indicates to the application that an ISOC OUT packet has been dropped. This bit does not have an associated mask bit and does not generate an interrupt. Dependency: This bit is valid in non Scatter/Gather DMA mode when periodic transfer interrupt feature is selected. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
9	bnaintr	BNA (Buffer Not Available) Interrupt (BNAIntr) This bit is valid only when Scatter/Gather DMA mode is enabled. The core generates this interrupt when the descriptor accessed is not ready For the Core to process, such as Host busy or DMA done **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
8	txfifoundrn	Fifo Underrun (TxfifoUndrn) Applies to IN endpoints Only This bit is valid only If thresholding is enabled. The core generates this interrupt when it detects a transmit FIFO underrun condition For this endpoint. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
7	txfemp	Transmit FIFO Empty (TxFEmp) This bit is valid only For IN Endpoints This interrupt is asserted when the TxFIFO For this endpoint is either half or completely empty. The half or completely empty status is determined by the TxFIFO Empty Level bit in the Core AHB Configuration register (GAHBCFG.NPTxFEmpLvl)). **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x1

Bit	Name	Description	Access	Reset
6	inepnakeff	IN Endpoint NAK Effective (INEPNakEff) Applies to periodic IN endpoints only. This bit can be cleared when the application clears the IN endpoint NAK by writing to DIEPCTLn.CNAK. This interrupt indicates that the core has sampled the NAK bit Set (either by the application or by the core). The interrupt indicates that the IN endpoint NAK bit Set by the application has taken effect in the core. This interrupt does not guarantee that a NAK handshake is sent on the USB. A STALL bit takes priority over a NAK bit. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
5	intknepmis	IN Token Received with EP Mismatch (INTknEPMis) Applies to non-periodic IN endpoints only. Indicates that the data in the top of the non-periodic TxFIFO belongs to an endpoint other than the one For which the IN token was received. This interrupt is asserted on the endpoint For which the IN token was received. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
4	intkntxfemp	IN Token Received When TxFIFO is Empty (INTknTXFEmp) Applies to non-periodic IN endpoints only. Indicates that an IN token was received when the associated TxFIFO (periodic/non-periodic) was empty. This interrupt is asserted on the endpoint For which the IN token was received. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
3	timeout	Timeout Condition (TimeOUT) In shared TX FIFO mode, applies to non-isochronous IN endpoints only. In dedicated FIFO mode, applies only to Control IN endpoints. In Scatter/Gather DMA mode, the TimeOUT interrupt is not asserted. Indicates that the core has detected a timeout condition on the USB For the last IN token on this endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
2	ahberr	AHB Error (AHBErr) Applies to IN and OUT endpoints. This is generated only in Internal DMA mode when there is an AHB error during an AHB read/write. The application can read the corresponding endpoint DMA address register to get the error address. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
1	epdisbld	Endpoint Disabled Interrupt (EPDisbld) Applies to IN and OUT endpoints. This bit indicates that the endpoint is disabled per the application's request. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
0	xfercompl	Transfer Completed Interrupt (XferCompl) Applies to IN and OUT endpoints. When Scatter/Gather DMA mode is enabled - For IN endpoint this field indicates that the requested data from the descriptor is moved from external system memory to internal FIFO. - For OUT endpoint this field indicates that the requested data from the internal FIFO is moved to external system memory. This interrupt is generated only when the corresponding endpoint descriptor is closed, and the IOC bit For the corresponding descriptor is Set. When Scatter/Gather DMA mode is disabled, this field indicates that the programmed transfer is complete on the AHB as well as on the USB, For this endpoint. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

dieptsiz5

Device IN Endpoint 5 Transfer Size Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB009B0
i_usbotg_1_devgrp	0xFFB40800	0xFFB409B0

Offset: 0x1B0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	mc RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
xfersize RW 0x0															

dieptsiz5 Fields

Bit	Name	Description	Access	Reset
30:29	mc	Applies to IN endpoints only. For periodic IN endpoints, this field indicates the number of packets that must be transmitted per microframe on the USB. The core uses this field to calculate the data PID for isochronous IN endpoints. 2'b01: 1 packet 2'b10: 2 packets 2'b11: 3 packets For non-periodic IN endpoints, this field is valid only in Internal DMA mode. It specifies the number of packets the core must fetchfor an IN endpoint before it switches to the endpoint pointed to by the Next Endpoint field of the Device Endpoint-n Control register (DIEPCTLn.NextEp) **Value** — **Description** 0x1 — PACKETONE 0x2 — PACKETTWO 0x3 — PACKETTHREE	RW	0x0

Bit	Name	Description	Access	Reset
28:19	pktcnt	Packet Count (PktCnt) Indicates the total number of USB packets that constitute the Transfer Size amount of data For endpoint 0. This field is decremented every time a packet (maximum size or short packet) is read from the TxFIFO.	RW	0x0
18:0	xfersize	Transfer Size (XferSize) Indicates the transfer size in bytes For endpoint 0. The core interrupts the application only after it has exhausted the transfer size amount of data. The transfer size can be Set to the maximum packet size of the endpoint, to be interrupted at the end of each packet. The core decrements this field every time a packet from the external memory is written to the TxFIFO.	RW	0x0

diepdma5

Device IN Endpoint 5 DMA Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB009B4
i_usbotg_1_devgrp	0xFFB40800	0xFFB409B4

Offset: 0x1B4

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepdma5 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
diepdma5 RW 0x0															

diepdma5 Fields

Bit	Name	Description	Access	Reset
31:0	diepdma5	Holds the start address of the external memory for storing or fetching endpoint data. Note: For control endpoints, this field stores control OUT data packets as well as SETUP transaction data packets. When more than three SETUP packets are received back-to-back, the SETUP data packet in the memory is overwritten. This register is incremented on every AHB transaction. The application can give only a DWORD-aligned address. When Scatter/Gather DMA mode is not enabled, the application programs the start address value in this field. When Scatter/Gather DMA mode is enabled, this field indicates the base pointer for the descriptor list.	RW	0x0

dtxfsts5

Device IN Endpoint Transmit FIFO Status Register 5

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB009B8
i_usbotg_1_devgrp	0xFFB40800	0xFFB409B8

Offset: 0x1B8

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ineptxfspcavail RO 0x2000															

dtxfsts5 Fields

Bit	Name	Description	Access	Reset
15:0	ineptxfspcavail	IN Endpoint TxFIFO Space Avail (INEPTxFSpcAvail) Indicates the amount of free space available in the Endpoint TxFIFO. Values are in terms of 32-bit words. 16'h0: Endpoint TxFIFO is full 16'h1: 1 word available 16'h2: 2 words available 16'hn: n words available (where 0 n 32,768) 16'h8000: 32,768 words available Others: Reserved	RO	0x2000

diepdmab5

Device IN Endpoint 5 Buffer Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB009BC
i_usbotg_1_devgrp	0xFFB40800	0xFFB409BC

Offset: 0x1BC

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepdmab5 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
diepdmab5 RO 0x0															

diepdmab5 Fields

Bit	Name	Description	Access	Reset
31:0	diepdmab5	Holds the current buffer address.This register is updated as and when the data transfer for the corresponding end point is in progress. This register is present only in Scatter/Gather DMA mode. Otherwise this field is reserved.	RO	0x0

diepctl6

Device Control IN Endpoint 6 Control Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB009C0
i_usbotg_1_devgrp	0xFFB40800	0xFFB409C0

Offset: 0x1C0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	txfnum RW 0x0				stall RW 0x0	Reserved	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
usbactep RW 0x0	Reserved				mps RW 0x0										

diepctl6 Fields

Bit	Name	Description	Access	Reset
31	epena	Endpoint Enable (EPEna) Applies to IN and OUT endpoints. When Scatter/Gather DMA mode is enabled, For IN endpoints this bit indicates that the descriptor structure and data buffer with data ready to transmit is setup. For OUT endpoint it indicates that the descriptor structure and data buffer to receive data is setup. When Scatter/Gather DMA mode is enabledsuch as for buffer-pointer based DMA mode: - For IN endpoints, this bit indicates that data is ready to be transmitted on the endpoint. - For OUT endpoints, this bit indicates that the application has allocated the memory to start receiving data from the USB. - The core clears this bit before setting any of the following interrupts on this endpoint: SETUP Phase Done Endpoint Disabled Transfer Completed Note: For control endpoints in DMA mode, this bit must be set to be able to transfer SETUP data packets in memory. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
30	epdis	Endpoint Disable (EPDis) Applies to IN and OUT endpoints. The application sets this bit to stop transmitting/receiving data on an endpoint, even before the transfer for that endpoint is complete. The application must wait for the Endpoint Disabled interrupt before treating the endpoint as disabled. The core clears this bit before setting the Endpoint Disabled interrupt. The application must set this bit only if Endpoint Enable is already set for this endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
29	setd1pid	Set DATA1 PID (SetD1PID) Applies to interrupt/bulk IN and OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA1. This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. Set Odd (micro)frame (SetOddFr) Applies to isochronous IN and OUT endpoints only. Writing to this field sets the Even/Odd (micro)frame (EO_FrNum) field to odd (micro)frame. This field is not applicable for Scatter/Gather DMA mode. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	WO	0x0

Bit	Name	Description	Access	Reset
28	setd0pid	Set DATA0 PID (SetD0PID) Applies to interrupt/bulk IN and OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA0. This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. 1'b0 WO In non-Scatter/Gather DMA mode: Set Even (micro)frame (SetEvenFr) Applies to isochronous IN and OUT endpoints only. Writing to this field sets the Even/Odd (micro)frame (EO_FrNum) field to even (micro) frame. When Scatter/Gather DMA mode is enabled, this field is reserved. The frame number in which to send data is in the transmit descriptor structure. The frame in which to receive data is updated in receive descriptor structure. **Value: Description** 0x0: DISABLED 0x1: ENABLED	WO	0x0
27	snak	Set NAK (SNAK) A write to this bit sets the NAK bit For the endpoint. Using this bit, the application can control the transmission of NAK handshakes on an endpoint. The core can also Set this bit For an endpoint after a SETUP packet is received on that endpoint. **Value: Description** 0x0: INACTIVE 0x1: ACTIVE	WO	0x0

Bit	Name	Description	Access	Reset
26	cnak	Clear NAK (CNAK) A write to this bit clears the NAK bit For the endpoint. Value: 0x0 — INACTIVE Value: 0x1 — ACTIVE	WO	0x0
25:22	txfnum	TxFIFO Number (TxFNum) Shared FIFO Operation non-periodic endpoints must set this bit to zero. Periodic endpoints must map this to the corresponding Periodic TxFIFO number. 4'h0: Non-Periodic TxFIFO Others: Specified Periodic TxFIFO.number Note: An interrupt IN endpoint can be configured as a non-periodic endpoint for applications such as mass storage. The core treats an IN endpoint as a non-periodic endpoint if the TxFNum field is set to 0. Otherwise, a separate periodic FIFO must be allocated for an interrupt IN endpoint, and the number of this FIFO must be programmed into the TxFNum field. Configuring an interrupt IN endpoint as a non-periodic endpoint saves the extra periodic FIFO area. Dedicated FIFO Operationthese bits specify the FIFO number associated with this endpoint. Each active IN endpoint must be programmed to a separate FIFO number. This field is valid only for IN endpoints.	RW	0x0

Bit	Name	Description	Access	Reset
21	stall	STALL Handshake (Stall) Applies to non-control, non-isochronous IN and OUT endpoints only. The application sets this bit to stall all tokens from the USB host to this endpoint. If a NAK bit, Global Non-periodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority. Only the application can clear this bit, never the core. 1'b0 R_W Applies to control endpoints only. The application can only set this bit, and the core clears it, when a SETUP token is received for this endpoint. If a NAK bit, Global Non-periodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority. Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake. **Value: Description** 0x0: INACTIVE 0x1: ACTIVE	RW	0x0
19:18	eptype	Endpoint Type (EPType) This is the transfer type supported by this logical endpoint. 2'b00: Control 2'b01: Isochronous 2'b10: Bulk 2'b11: Interrupt **Value: Description** 0x0: CONTROL 0x1: ISOCHRONOUS 0x2: BULK 0x3: INTERRUP	RW	0x0

Bit	Name	Description	Access	Reset
17	naksts	NAK Status (NAKSts) Indicates the following: 1'b0: The core is transmitting non-NAK handshakes based on the FIFO status. 1'b1: The core is transmitting NAK handshakes on this endpoint. When either the application or the core sets this bit: The core stops receiving any data on an OUT endpoint, even if there is space in the RxFIFO to accommodate the incoming packet. For non-isochronous IN endpoints: The core stops transmitting any data on an IN endpoint, even if there data is available in the TxFIFO. For isochronous IN endpoints: The core sends out a zero-length data packet, even if there data is available in the TxFIFO. Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake. **Value** — **Description** 0x0 — NONNAK 0x1 — NAK	RO	0x0

Bit	Name	Description	Access	Reset
16	dpid	Endpoint Data PID (DPID) Applies to interrupt/bulk IN and OUT endpoints only. Contains the PID of the packet to be received or transmitted on this endpoint. The application must program the PID of the first packet to be received or transmitted on this endpoint, after the endpoint is activated. The applications use the SetD1PID and SetD0PID fields of this register to program either DATA0 or DATA1 PID. 1'b0: DATA0 1'b1: DATA1 This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. 1'b0 RO Even/Odd (Micro)Frame (EO_FrNum) In non-Scatter/Gather DMA mode: Applies to isochronous IN and OUT endpoints only. Indicates the (micro)frame number in which the core transmits/receives isochronous data for this endpoint. The application must program the even/odd (micro) frame number in which it intends to transmit/receive isochronous data for this endpoint using the SetEvnFr and SetOddFr fields in this register. 1'b0: Even (micro)frame 1'b1: Odd (micro)frame When Scatter/Gather DMA mode is enabled, this field is reserved. The frame number in which to send data is provided in the transmit descriptor structure. The frame in which data is received is updated in receive descriptor structure. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RO	0x0

Bit	Name	Description	Access	Reset
15	usbactep	USB Active Endpoint (USBActEP) Indicates whether this endpoint is active in the current configuration and interface. The core clears this bit for all endpoints (other than EP 0) after detecting a USB reset. After receiving the SetConfiguration and SetInterface commands, the applica-tion must program endpoint registers accordingly and set this bit. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
10:0	mps	Maximum Packet Size (MPS) The application must program this field with the maximum packet size for the current logical endpoint. This value is in bytes.	RW	0x0

diepint6

Device IN Endpoint 6 Interrupt Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB009C8
i_usbotg_1_devgrp	0xFFB40800	0xFFB409C8

Offset: 0x1C8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	txfifoundrn RW 0x0	txfempp RO 0x1	inepnakeff RW 0x0	intknepmis RW 0x0	intkntxfempp RW 0x0	timeout RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0

diepint6 Fields

Bit	Name	Description	Access	Reset
14	nyetintrpt	NYET Interrupt (NYETIntrpt) The core generates this interrupt when a NYET response is transmitted for a non isochronous OUT endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
13	nakintrpt	NAK Interrupt (NAKInterrupt) The core generates this interrupt when a NAK is transmitted or received by the device. In case of isochronous IN endpoints the interrupt gets generated when a zero length packet is transmitted due to un-availability of data in the TXFifo. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
12	bbleerr	NAK Interrupt (BbleErr) The core generates this interrupt when babble is received for the endpoint. **Value** / **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
11	pktdrpsts	Packet Drop Status (PktDrpSts) This bit indicates to the application that an ISOC OUT packet has been dropped. This bit does not have an associated mask bit and does not generate an interrupt. Dependency: This bit is valid in non Scatter/Gather DMA mode when periodic transfer interrupt feature is selected. **Value** / **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
9	bnaintr	BNA (Buffer Not Available) Interrupt (BNAIntr) This bit is valid only when Scatter/Gather DMA mode is enabled. The core generates this interrupt when the descriptor accessed is not ready For the Core to process, such as Host busy or DMA done **Value** / **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
8	txfifoundrn	Fifo Underrun (TxfifoUndrn) Applies to IN endpoints Only This bit is valid only If thresholding is enabled. The core generates this interrupt when it detects a transmit FIFO underrun condition For this endpoint. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
7	txfemp	Transmit FIFO Empty (TxFEmp) This bit is valid only For IN Endpoints This interrupt is asserted when the TxFIFO For this endpoint is either half or completely empty. The half or completely empty status is determined by the TxFIFO Empty Level bit in the Core AHB Configuration register (GAHBCFG.NPTxFEmpLvl)). **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x1

Bit	Name	Description	Access	Reset
6	inepnakeff	IN Endpoint NAK Effective (INEPNakEff) Applies to periodic IN endpoints only. This bit can be cleared when the application clears the IN endpoint NAK by writing to DIEPCTLn.CNAK. This interrupt indicates that the core has sampled the NAK bit Set (either by the application or by the core). The interrupt indicates that the IN endpoint NAK bit Set by the application has taken effect in the core. This interrupt does not guarantee that a NAK handshake is sent on the USB. A STALL bit takes priority over a NAK bit. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
5	intknepmis	IN Token Received with EP Mismatch (INTknEPMis) Applies to non-periodic IN endpoints only. Indicates that the data in the top of the non-periodic TxFIFO belongs to an endpoint other than the one For which the IN token was received. This interrupt is asserted on the endpoint For which the IN token was received. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
4	intkntxfemp	IN Token Received When TxFIFO is Empty (INTknTXFEmp) Applies to non-periodic IN endpoints only. Indicates that an IN token was received when the associated TxFIFO (periodic/non-periodic) was empty. This interrupt is asserted on the endpoint For which the IN token was received. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
3	timeout	Timeout Condition (TimeOUT) In shared TX FIFO mode, applies to non-isochronous IN endpoints only. In dedicated FIFO mode, applies only to Control IN endpoints. In Scatter/Gather DMA mode, the TimeOUT interrupt is not asserted. Indicates that the core has detected a timeout condition on the USB For the last IN token on this endpoint. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
2	ahberr	AHB Error (AHBErr) Applies to IN and OUT endpoints. This is generated only in Internal DMA mode when there is an AHB error during an AHB read/write. The application can read the corresponding endpoint DMA address register to get the error address. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
1	epdisbld	Endpoint Disabled Interrupt (EPDisbld) Applies to IN and OUT endpoints. This bit indicates that the endpoint is disabled per the application's request. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
0	xfercompl	Transfer Completed Interrupt (XferCompl) Applies to IN and OUT endpoints. When Scatter/Gather DMA mode is enabled - For IN endpoint this field indicates that the requested data from the descriptor is moved from external system memory to internal FIFO. - For OUT endpoint this field indicates that the requested data from the internal FIFO is moved to external system memory. This interrupt is generated only when the corresponding endpoint descriptor is closed, and the IOC bit For the corresponding descriptor is Set. When Scatter/Gather DMA mode is disabled, this field indicates that the programmed transfer is complete on the AHB as well as on the USB, For this endpoint. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

dieptsiz6

Device IN Endpoint 6 Transfer Size Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB009D0
i_usbotg_1_devgrp	0xFFB40800	0xFFB409D0

Offset: 0x1D0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	mc RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
xfersize RW 0x0															

dieptsiz6 Fields

Bit	Name	Description	Access	Reset
30:29	mc	Applies to IN endpoints only. For periodic IN endpoints, this field indicates the number of packets that must be transmitted per microframe on the USB. The core uses this field to calculate the data PID for isochronous IN endpoints. 2'b01: 1 packet 2'b10: 2 packets 2'b11: 3 packets For non-periodic IN endpoints, this field is valid only in Internal DMA mode. It specifies the number of packets the core must fetchfor an IN endpoint before it switches to the endpoint pointed to by the Next Endpoint field of the Device Endpoint-n Control register (DIEPCTLn.NextEp) **Value** — **Description** 0x1 — PACKETONE 0x2 — PACKETTWO 0x3 — PACKETTHREE	RW	0x0

Bit	Name	Description	Access	Reset
28:19	pktcnt	Packet Count (PktCnt) Indicates the total number of USB packets that constitute the Transfer Size amount of data For endpoint 0. This field is decremented every time a packet (maximum size or short packet) is read from the TxFIFO.	RW	0x0
18:0	xfersize	Transfer Size (XferSize) Indicates the transfer size in bytes For endpoint 0. The core interrupts the application only after it has exhausted the transfer size amount of data. The transfer size can be Set to the maximum packet size of the endpoint, to be interrupted at the end of each packet. The core decrements this field every time a packet from the external memory is written to the TxFIFO.	RW	0x0

diepdma6

Device IN Endpoint 6 DMA Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB009D4
i_usbotg_1_devgrp	0xFFB40800	0xFFB409D4

Offset: 0x1D4

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepdma6 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
diepdma6 RW 0x0															

diepdma6 Fields

Bit	Name	Description	Access	Reset
31:0	diepdma6	Holds the start address of the external memory for storing or fetching endpoint data. Note: For control endpoints, this field stores control OUT data packets as well as SETUP transaction data packets. When more than three SETUP packets are received back-to-back, the SETUP data packet in the memory is overwritten. This register is incremented on every AHB transaction. The application can give only a DWORD-aligned address. When Scatter/Gather DMA mode is not enabled, the application programs the start address value in this field. When Scatter/Gather DMA mode is enabled, this field indicates the base pointer for the descriptor list.	RW	0x0

dtxfsts6

Device IN Endpoint Transmit FIFO Status Register 6

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB009D8
i_usbotg_1_devgrp	0xFFB40800	0xFFB409D8

Offset: 0x1D8

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ineptxfspcavail RO 0x2000															

dtxfsts6 Fields

Bit	Name	Description	Access	Reset
15:0	ineptxfspcavail	IN Endpoint TxFIFO Space Avail (INEPTxFSpcAvail) Indicates the amount of free space available in the Endpoint TxFIFO. Values are in terms of 32-bit words. 16'h0: Endpoint TxFIFO is full 16'h1: 1 word available 16'h2: 2 words available 16'hn: n words available (where 0 n 32,768) 16'h8000: 32,768 words available Others: Reserved	RO	0x2000

diepdmab6

Device IN Endpoint 6 Buffer Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB009DC
i_usbotg_1_devgrp	0xFFB40800	0xFFB409DC

Offset: 0x1DC

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepdmab6 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
diepdmab6 RO 0x0															

diepdmab6 Fields

Bit	Name	Description	Access	Reset
31:0	diepdmab6	Holds the current buffer address.This register is updated as and when the data transfer for the corresponding end point is in progress. This register is present only in Scatter/Gather DMA mode. Otherwise this field is reserved.	RO	0x0

diepctl7

Device Control IN Endpoint 7 Control Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB009E0
i_usbotg_1_devgrp	0xFFB40800	0xFFB409E0

Offset: 0x1E0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	txfnum RW 0x0				stall RW 0x0	Reserved	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
usbactep RW 0x0	Reserved				mps RW 0x0										

diepctl7 Fields

Bit	Name	Description	Access	Reset
31	epena	Endpoint Enable (EPEna) Applies to IN and OUT endpoints. When Scatter/Gather DMA mode is enabled, For IN endpoints this bit indicates that the descriptor structure and data buffer with data ready to transmit is setup. For OUT endpoint it indicates that the descriptor structure and data buffer to receive data is setup. When Scatter/Gather DMA mode is enabledsuch as for buffer-pointer based DMA mode: - For IN endpoints, this bit indicates that data is ready to be transmitted on the endpoint. - For OUT endpoints, this bit indicates that the application has allocated the memory to start receiving data from the USB. - The core clears this bit before setting any of the following interrupts on this endpoint: SETUP Phase Done Endpoint Disabled Transfer Completed Note: For control endpoints in DMA mode, this bit must be set to be able to transfer SETUP data packets in memory. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
30	epdis	Endpoint Disable (EPDis) Applies to IN and OUT endpoints. The application sets this bit to stop transmitting/receiving data on an endpoint, even before the transfer for that endpoint is complete. The application must wait for the Endpoint Disabled interrupt before treating the endpoint as disabled. The core clears this bit before setting the Endpoint Disabled interrupt. The application must set this bit only if Endpoint Enable is already set for this endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
29	setd1pid	Set DATA1 PID (SetD1PID) Applies to interrupt/bulk IN and OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA1. This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. Set Odd (micro)frame (SetOddFr) Applies to isochronous IN and OUT endpoints only. Writing to this field sets the Even/Odd (micro)frame (EO_FrNum) field to odd (micro)frame. This field is not applicable for Scatter/Gather DMA mode. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	WO	0x0

Bit	Name	Description	Access	Reset
28	setd0pid	Set DATA0 PID (SetD0PID) Applies to interrupt/bulk IN and OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA0. This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. 1'b0 WO In non-Scatter/Gather DMA mode: Set Even (micro)frame (SetEvenFr) Applies to isochronous IN and OUT endpoints only. Writing to this field sets the Even/Odd (micro)frame (EO_FrNum) field to even (micro) frame. When Scatter/Gather DMA mode is enabled, this field is reserved. The frame number in which to send data is in the transmit descriptor structure. The frame in which to receive data is updated in receive descriptor structure. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	WO	0x0
27	snak	Set NAK (SNAK) A write to this bit sets the NAK bit For the endpoint. Using this bit, the application can control the transmission of NAK handshakes on an endpoint. The core can also Set this bit For an endpoint after a SETUP packet is received on that endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	WO	0x0

Bit	Name	Description	Access	Reset
26	cnak	Clear NAK (CNAK) A write to this bit clears the NAK bit For the endpoint. Value: 0x0 — INACTIVE Value: 0x1 — ACTIVE	WO	0x0
25:22	txfnum	TxFIFO Number (TxFNum) Shared FIFO Operation non-periodic endpoints must set this bit to zero. Periodic endpoints must map this to the corresponding Periodic TxFIFO number. 4'h0: Non-Periodic TxFIFO Others: Specified Periodic TxFIFO.number Note: An interrupt IN endpoint can be configured as a non-periodic endpoint for applications such as mass storage. The core treats an IN endpoint as a non-periodic endpoint if the TxFNum field is set to 0. Otherwise, a separate periodic FIFO must be allocated for an interrupt IN endpoint, and the number of this FIFO must be programmed into the TxFNum field. Configuring an interrupt IN endpoint as a non-periodic endpoint saves the extra periodic FIFO area. Dedicated FIFO Operationthese bits specify the FIFO number associated with this endpoint. Each active IN endpoint must be programmed to a separate FIFO number. This field is valid only for IN endpoints.	RW	0x0

Bit	Name	Description	Access	Reset
21	stall	STALL Handshake (Stall) Applies to non-control, non-isochronous IN and OUT endpoints only. The application sets this bit to stall all tokens from the USB host to this endpoint. If a NAK bit, Global Non-periodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority. Only the application can clear this bit, never the core. 1'b0 R_W Applies to control endpoints only. The application can only set this bit, and the core clears it, when a SETUP token is received for this endpoint. If a NAK bit, Global Non-periodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority. Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake. **Value: Description** 0x0: INACTIVE 0x1: ACTIVE	RW	0x0
19:18	eptype	Endpoint Type (EPType) This is the transfer type supported by this logical endpoint. 2'b00: Control 2'b01: Isochronous 2'b10: Bulk 2'b11: Interrupt **Value: Description** 0x0: CONTROL 0x1: ISOCHRONOUS 0x2: BULK 0x3: INTERRUP	RW	0x0

Bit	Name	Description	Access	Reset
17	naksts	NAK Status (NAKSts) Indicates the following: 1'b0: The core is transmitting non-NAK handshakes based on the FIFO status. 1'b1: The core is transmitting NAK handshakes on this endpoint. When either the application or the core sets this bit: The core stops receiving any data on an OUT endpoint, even if there is space in the RxFIFO to accommodate the incoming packet. For non-isochronous IN endpoints: The core stops transmitting any data on an IN endpoint, even if there data is available in the TxFIFO. For isochronous IN endpoints: The core sends out a zero-length data packet, even if there data is available in the TxFIFO. Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake. **Value** — **Description** 0x0 — NONNAK 0x1 — NAK	RO	0x0

Bit	Name	Description	Access	Reset
16	dpid	Endpoint Data PID (DPID) Applies to interrupt/bulk IN and OUT endpoints only. Contains the PID of the packet to be received or transmitted on this endpoint. The application must program the PID of the first packet to be received or transmitted on this endpoint, after the endpoint is activated. The applications use the SetD1PID and SetD0PID fields of this register to program either DATA0 or DATA1 PID. 1'b0: DATA0 1'b1: DATA1 This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. 1'b0 RO Even/Odd (Micro)Frame (EO_FrNum) In non-Scatter/Gather DMA mode: Applies to isochronous IN and OUT endpoints only. Indicates the (micro)frame number in which the core transmits/receives isochronous data for this endpoint. The application must program the even/odd (micro) frame number in which it intends to transmit/receive isochronous data for this endpoint using the SetEvnFr and SetOddFr fields in this register. 1'b0: Even (micro)frame 1'b1: Odd (micro)frame When Scatter/Gather DMA mode is enabled, this field is reserved. The frame number in which to send data is provided in the transmit descriptor structure. The frame in which data is received is updated in receive descriptor structure. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RO	0x0

Bit	Name	Description	Access	Reset
15	usbactep	USB Active Endpoint (USBActEP) Indicates whether this endpoint is active in the current configuration and interface. The core clears this bit for all endpoints (other than EP 0) after detecting a USB reset. After receiving the SetConfiguration and SetInterface commands, the applica-tion must program endpoint registers accordingly and set this bit. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
10:0	mps	Maximum Packet Size (MPS) The application must program this field with the maximum packet size for the current logical endpoint. This value is in bytes.	RW	0x0

diepint7

Device IN Endpoint 7 Interrupt Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB009E8
i_usbotg_1_devgrp	0xFFB40800	0xFFB409E8

Offset: 0x1E8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	txfifoundrn RW 0x0	txfempp RO 0x1	inepnakeff RW 0x0	intknepmis RW 0x0	intkntxfempp RW 0x0	timeout RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0

diepint7 Fields

Bit	Name	Description	Access	Reset
14	nyetintrpt	NYET Interrupt (NYETIntrpt) The core generates this interrupt when a NYET response is transmitted for a non isochronous OUT endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
13	nakintrpt	NAK Interrupt (NAKInterrupt) The core generates this interrupt when a NAK is transmitted or received by the device. In case of isochronous IN endpoints the interrupt gets generated when a zero length packet is transmitted due to un-availability of data in the TXFifo. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
12	bbleerr	NAK Interrupt (BbleErr) The core generates this interrupt when babble is received for the endpoint. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
11	pktdrpsts	Packet Drop Status (PktDrpSts) This bit indicates to the application that an ISOC OUT packet has been dropped. This bit does not have an associated mask bit and does not generate an interrupt. Dependency: This bit is valid in non Scatter/Gather DMA mode when periodic transfer interrupt feature is selected. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
9	bnaintr	BNA (Buffer Not Available) Interrupt (BNAIntr) This bit is valid only when Scatter/Gather DMA mode is enabled. The core generates this interrupt when the descriptor accessed is not ready For the Core to process, such as Host busy or DMA done **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
8	txfifoundrn	Fifo Underrun (TxfifoUndrn) Applies to IN endpoints Only This bit is valid only If thresholding is enabled. The core generates this interrupt when it detects a transmit FIFO underrun condition For this endpoint. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
7	txfemp	Transmit FIFO Empty (TxFEmp) This bit is valid only For IN Endpoints This interrupt is asserted when the TxFIFO For this endpoint is either half or completely empty. The half or completely empty status is determined by the TxFIFO Empty Level bit in the Core AHB Configuration register (GAHBCFG.NPTxFEmpLvl)). **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x1

Bit	Name	Description	Access	Reset
6	inepnakeff	IN Endpoint NAK Effective (INEPNakEff) Applies to periodic IN endpoints only. This bit can be cleared when the application clears the IN endpoint NAK by writing to DIEPCTLn.CNAK. This interrupt indicates that the core has sampled the NAK bit Set (either by the application or by the core). The interrupt indicates that the IN endpoint NAK bit Set by the application has taken effect in the core. This interrupt does not guarantee that a NAK handshake is sent on the USB. A STALL bit takes priority over a NAK bit. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
5	intknepmis	IN Token Received with EP Mismatch (INTknEPMis) Applies to non-periodic IN endpoints only. Indicates that the data in the top of the non-periodic TxFIFO belongs to an endpoint other than the one For which the IN token was received. This interrupt is asserted on the endpoint For which the IN token was received. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
4	intkntxfemp	IN Token Received When TxFIFO is Empty (INTknTXFEmp) Applies to non-periodic IN endpoints only. Indicates that an IN token was received when the associated TxFIFO (periodic/non-periodic) was empty. This interrupt is asserted on the endpoint For which the IN token was received. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
3	timeout	Timeout Condition (TimeOUT) In shared TX FIFO mode, applies to non-isochronous IN endpoints only. In dedicated FIFO mode, applies only to Control IN endpoints. In Scatter/Gather DMA mode, the TimeOUT interrupt is not asserted. Indicates that the core has detected a timeout condition on the USB For the last IN token on this endpoint. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
2	ahberr	AHB Error (AHBErr) Applies to IN and OUT endpoints. This is generated only in Internal DMA mode when there is an AHB error during an AHB read/write. The application can read the corresponding endpoint DMA address register to get the error address. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
1	epdisbld	Endpoint Disabled Interrupt (EPDisbld) Applies to IN and OUT endpoints. This bit indicates that the endpoint is disabled per the application's request. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
0	xfercompl	Transfer Completed Interrupt (XferCompl) Applies to IN and OUT endpoints. When Scatter/Gather DMA mode is enabled - For IN endpoint this field indicates that the requested data from the descriptor is moved from external system memory to internal FIFO. - For OUT endpoint this field indicates that the requested data from the internal FIFO is moved to external system memory. This interrupt is generated only when the corresponding endpoint descriptor is closed, and the IOC bit For the corresponding descriptor is Set. When Scatter/Gather DMA mode is disabled, this field indicates that the programmed transfer is complete on the AHB as well as on the USB, For this endpoint. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

dieptsiz7

Device IN Endpoint 7 Transfer Size Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB009F0
i_usbotg_1_devgrp	0xFFB40800	0xFFB409F0

Offset: 0x1F0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	mc RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
xfersize RW 0x0															

dieptsiz7 Fields

Bit	Name	Description	Access	Reset
30:29	mc	Applies to IN endpoints only. For periodic IN endpoints, this field indicates the number of packets that must be transmitted per microframe on the USB. The core uses this field to calculate the data PID for isochronous IN endpoints. 2'b01: 1 packet 2'b10: 2 packets 2'b11: 3 packets For non-periodic IN endpoints, this field is valid only in Internal DMA mode. It specifies the number of packets the core must fetchfor an IN endpoint before it switches to the endpoint pointed to by the Next Endpoint field of the Device Endpoint-n Control register (DIEPCTLn.NextEp) **Value** — **Description** 0x1 — PACKETONE 0x2 — PACKETTWO 0x3 — PACKETTHREE	RW	0x0

Bit	Name	Description	Access	Reset
28:19	pktcnt	Packet Count (PktCnt) Indicates the total number of USB packets that constitute the Transfer Size amount of data For endpoint 0. This field is decremented every time a packet (maximum size or short packet) is read from the TxFIFO.	RW	0x0
18:0	xfersize	Transfer Size (XferSize) Indicates the transfer size in bytes For endpoint 0. The core interrupts the application only after it has exhausted the transfer size amount of data. The transfer size can be Set to the maximum packet size of the endpoint, to be interrupted at the end of each packet. The core decrements this field every time a packet from the external memory is written to the TxFIFO.	RW	0x0

diepdma7

Device IN Endpoint 7 DMA Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB009F4
i_usbotg_1_devgrp	0xFFB40800	0xFFB409F4

Offset: 0x1F4

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepdma7 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
diepdma7 RW 0x0															

diepdma7 Fields

Bit	Name	Description	Access	Reset
31:0	diepdma7	Holds the start address of the external memory for storing or fetching endpoint data. Note: For control endpoints, this field stores control OUT data packets as well as SETUP transaction data packets. When more than three SETUP packets are received back-to-back, the SETUP data packet in the memory is overwritten. This register is incremented on every AHB transaction. The application can give only a DWORD-aligned address. When Scatter/Gather DMA mode is not enabled, the application programs the start address value in this field. When Scatter/Gather DMA mode is enabled, this field indicates the base pointer for the descriptor list.	RW	0x0

dtxfsts7

Device IN Endpoint Transmit FIFO Status Register 7

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB009F8
i_usbotg_1_devgrp	0xFFB40800	0xFFB409F8

Offset: 0x1F8

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ineptxfspcavail RO 0x2000															

dtxfsts7 Fields

Bit	Name	Description	Access	Reset
15:0	ineptxfspcavail	IN Endpoint TxFIFO Space Avail (INEPTxFSpcAvail) Indicates the amount of free space available in the Endpoint TxFIFO. Values are in terms of 32-bit words. 16'h0: Endpoint TxFIFO is full 16'h1: 1 word available 16'h2: 2 words available 16'hn: n words available (where 0 n 32,768) 16'h8000: 32,768 words available Others: Reserved	RO	0x2000

diepdmab7

Device IN Endpoint 7 Buffer Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB009FC
i_usbotg_1_devgrp	0xFFB40800	0xFFB409FC

Offset: 0x1FC

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepdmab7 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
diepdmab7 RO 0x0															

diepdmab7 Fields

Bit	Name	Description	Access	Reset
31:0	diepdmab7	Holds the current buffer address.This register is updated as and when the data transfer for the corresponding end point is in progress. This register is present only in Scatter/Gather DMA mode. Otherwise this field is reserved.	RO	0x0

diepctl8

Device Control IN Endpoint 8 Control Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00A00
i_usbotg_1_devgrp	0xFFB40800	0xFFB40A00

Offset: 0x200

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	txfnum RW 0x0				stall RW 0x0	Reserved	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
usbactep RW 0x0	Reserved				mps RW 0x0										

diepctl8 Fields

Bit	Name	Description	Access	Reset
31	epena	Endpoint Enable (EPEna) Applies to IN and OUT endpoints. When Scatter/Gather DMA mode is enabled, For IN endpoints this bit indicates that the descriptor structure and data buffer with data ready to transmit is setup. For OUT endpoint it indicates that the descriptor structure and data buffer to receive data is setup. When Scatter/Gather DMA mode is enabledsuch as for buffer-pointer based DMA mode: - For IN endpoints, this bit indicates that data is ready to be transmitted on the endpoint. - For OUT endpoints, this bit indicates that the application has allocated the memory to start receiving data from the USB. - The core clears this bit before setting any of the following interrupts on this endpoint: SETUP Phase Done Endpoint Disabled Transfer Completed Note: For control endpoints in DMA mode, this bit must be set to be able to transfer SETUP data packets in memory. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
30	epdis	Endpoint Disable (EPDis) Applies to IN and OUT endpoints. The application sets this bit to stop transmitting/receiving data on an endpoint, even before the transfer for that endpoint is complete. The application must wait for the Endpoint Disabled interrupt before treating the endpoint as disabled. The core clears this bit before setting the Endpoint Disabled interrupt. The application must set this bit only if Endpoint Enable is already set for this endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
29	setd1pid	Set DATA1 PID (SetD1PID) Applies to interrupt/bulk IN and OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA1. This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. Set Odd (micro)frame (SetOddFr) Applies to isochronous IN and OUT endpoints only. Writing to this field sets the Even/Odd (micro)frame (EO_FrNum) field to odd (micro)frame. This field is not applicable for Scatter/Gather DMA mode. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	WO	0x0

Bit	Name	Description	Access	Reset
28	setd0pid	Set DATA0 PID (SetD0PID) Applies to interrupt/bulk IN and OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA0. This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. 1'b0 WO In non-Scatter/Gather DMA mode: Set Even (micro)frame (SetEvenFr) Applies to isochronous IN and OUT endpoints only. Writing to this field sets the Even/Odd (micro)frame (EO_FrNum) field to even (micro) frame. When Scatter/Gather DMA mode is enabled, this field is reserved. The frame number in which to send data is in the transmit descriptor structure. The frame in which to receive data is updated in receive descriptor structure. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	WO	0x0
27	snak	Set NAK (SNAK) A write to this bit sets the NAK bit For the endpoint. Using this bit, the application can control the transmission of NAK handshakes on an endpoint. The core can also Set this bit For an endpoint after a SETUP packet is received on that endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	WO	0x0

Bit	Name	Description	Access	Reset
26	cnak	Clear NAK (CNAK) A write to this bit clears the NAK bit For the endpoint. Value: 0x0 — Description: INACTIVE Value: 0x1 — Description: ACTIVE	WO	0x0
25:22	txfnum	TxFIFO Number (TxFNum) Shared FIFO Operation non-periodic endpoints must set this bit to zero. Periodic endpoints must map this to the corresponding Periodic TxFIFO number. 4'h0: Non-Periodic TxFIFO Others: Specified Periodic TxFIFO.number Note: An interrupt IN endpoint can be configured as a non-periodic endpoint for applications such as mass storage. The core treats an IN endpoint as a non-periodic endpoint if the TxFNum field is set to 0. Otherwise, a separate periodic FIFO must be allocated for an interrupt IN endpoint, and the number of this FIFO must be programmed into the TxFNum field. Configuring an interrupt IN endpoint as a non-periodic endpoint saves the extra periodic FIFO area. Dedicated FIFO Operationthese bits specify the FIFO number associated with this endpoint. Each active IN endpoint must be programmed to a separate FIFO number. This field is valid only for IN endpoints.	RW	0x0

Bit	Name	Description	Access	Reset
21	stall	STALL Handshake (Stall) Applies to non-control, non-isochronous IN and OUT endpoints only. The application sets this bit to stall all tokens from the USB host to this endpoint. If a NAK bit, Global Non-periodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority. Only the application can clear this bit, never the core. 1'b0 R_W Applies to control endpoints only. The application can only set this bit, and the core clears it, when a SETUP token is received for this endpoint. If a NAK bit, Global Non-periodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority. Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake. **Value: Description** 0x0: INACTIVE 0x1: ACTIVE	RW	0x0
19:18	eptype	Endpoint Type (EPType) This is the transfer type supported by this logical endpoint. 2'b00: Control 2'b01: Isochronous 2'b10: Bulk 2'b11: Interrupt **Value: Description** 0x0: CONTROL 0x1: ISOCHRONOUS 0x2: BULK 0x3: INTERRUP	RW	0x0

Bit	Name	Description	Access	Reset
17	naksts	NAK Status (NAKSts) Indicates the following: 1'b0: The core is transmitting non-NAK handshakes based on the FIFO status. 1'b1: The core is transmitting NAK handshakes on this endpoint. When either the application or the core sets this bit: The core stops receiving any data on an OUT endpoint, even if there is space in the RxFIFO to accommodate the incoming packet. For non-isochronous IN endpoints: The core stops transmitting any data on an IN endpoint, even if there data is available in the TxFIFO. For isochronous IN endpoints: The core sends out a zero-length data packet, even if there data is available in the TxFIFO. Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake. **Value** — **Description** 0x0 — NONNAK 0x1 — NAK	RO	0x0

Bit	Name	Description	Access	Reset
16	dpid	Endpoint Data PID (DPID) Applies to interrupt/bulk IN and OUT endpoints only. Contains the PID of the packet to be received or transmitted on this endpoint. The application must program the PID of the first packet to be received or transmitted on this endpoint, after the endpoint is activated. The applications use the SetD1PID and SetD0PID fields of this register to program either DATA0 or DATA1 PID. 1'b0: DATA0 1'b1: DATA1 This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. 1'b0 RO Even/Odd (Micro)Frame (EO_FrNum) In non-Scatter/Gather DMA mode: Applies to isochronous IN and OUT endpoints only. Indicates the (micro)frame number in which the core transmits/ receives isochronous data for this endpoint. The application must program the even/odd (micro) frame number in which it intends to transmit/receive isochronous data for this endpoint using the SetEvnFr and SetOddFr fields in this register. 1'b0: Even (micro)frame 1'b1: Odd (micro)frame When Scatter/Gather DMA mode is enabled, this field is reserved. The frame number in which to send data is provided in the transmit descriptor structure. The frame in which data is received is updated in receive descriptor structure. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RO	0x0

Bit	Name	Description	Access	Reset
15	usbactep	USB Active Endpoint (USBActEP) Indicates whether this endpoint is active in the current configuration and interface. The core clears this bit for all endpoints (other than EP 0) after detecting a USB reset. After receiving the SetConfiguration and SetInterface commands, the applica-tion must program endpoint registers accordingly and set this bit. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
10:0	mps	Maximum Packet Size (MPS) The application must program this field with the maximum packet size for the current logical endpoint. This value is in bytes.	RW	0x0

diepint8

Device IN Endpoint 8 Interrupt Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00A08
i_usbotg_1_devgrp	0xFFB40800	0xFFB40A08

Offset: 0x208

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	txfifoundrn RW 0x0	txfempp RO 0x1	inepnakeff RW 0x0	intknepmis RW 0x0	intkntxfempp RW 0x0	timeout RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0

diepint8 Fields

Bit	Name	Description	Access	Reset
14	nyetintrpt	NYET Interrupt (NYETIntrpt) The core generates this interrupt when a NYET response is transmitted for a non isochronous OUT endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
13	nakintrpt	NAK Interrupt (NAKInterrupt) The core generates this interrupt when a NAK is transmitted or received by the device. In case of isochronous IN endpoints the interrupt gets generated when a zero length packet is transmitted due to un-availability of data in the TXFifo. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
12	bbleerr	NAK Interrupt (BbleErr) The core generates this interrupt when babble is received for the endpoint. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
11	pktdrpsts	Packet Drop Status (PktDrpSts) This bit indicates to the application that an ISOC OUT packet has been dropped. This bit does not have an associated mask bit and does not generate an interrupt. Dependency: This bit is valid in non Scatter/Gather DMA mode when periodic transfer interrupt feature is selected. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
9	bnaintr	BNA (Buffer Not Available) Interrupt (BNAIntr) This bit is valid only when Scatter/Gather DMA mode is enabled. The core generates this interrupt when the descriptor accessed is not ready For the Core to process, such as Host busy or DMA done **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
8	txfifoundrn	Fifo Underrun (TxfifoUndrn) Applies to IN endpoints Only This bit is valid only If thresholding is enabled. The core generates this interrupt when it detects a transmit FIFO underrun condition For this endpoint. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
7	txfemp	Transmit FIFO Empty (TxFEmp) This bit is valid only For IN Endpoints This interrupt is asserted when the TxFIFO For this endpoint is either half or completely empty. The half or completely empty status is determined by the TxFIFO Empty Level bit in the Core AHB Configuration register (GAHBCFG.NPTxFEmpLvl)). **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x1

Bit	Name	Description	Access	Reset
6	inepnakeff	IN Endpoint NAK Effective (INEPNakEff) Applies to periodic IN endpoints only. This bit can be cleared when the application clears the IN endpoint NAK by writing to DIEPCTLn.CNAK. This interrupt indicates that the core has sampled the NAK bit Set (either by the application or by the core). The interrupt indicates that the IN endpoint NAK bit Set by the application has taken effect in the core. This interrupt does not guarantee that a NAK handshake is sent on the USB. A STALL bit takes priority over a NAK bit. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
5	intknepmis	IN Token Received with EP Mismatch (INTknEPMis) Applies to non-periodic IN endpoints only. Indicates that the data in the top of the non-periodic TxFIFO belongs to an endpoint other than the one For which the IN token was received. This interrupt is asserted on the endpoint For which the IN token was received. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
4	intkntxfemp	IN Token Received When TxFIFO is Empty (INTknTXFEmp) Applies to non-periodic IN endpoints only. Indicates that an IN token was received when the associated TxFIFO (periodic/non-periodic) was empty. This interrupt is asserted on the endpoint For which the IN token was received. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
3	timeout	Timeout Condition (TimeOUT) In shared TX FIFO mode, applies to non-isochronous IN endpoints only. In dedicated FIFO mode, applies only to Control IN endpoints. In Scatter/Gather DMA mode, the TimeOUT interrupt is not asserted. Indicates that the core has detected a timeout condition on the USB For the last IN token on this endpoint. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
2	ahberr	AHB Error (AHBErr) Applies to IN and OUT endpoints. This is generated only in Internal DMA mode when there is an AHB error during an AHB read/write. The application can read the corresponding endpoint DMA address register to get the error address. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
1	epdisbld	Endpoint Disabled Interrupt (EPDisbld) Applies to IN and OUT endpoints. This bit indicates that the endpoint is disabled per the application's request. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
0	xfercompl	Transfer Completed Interrupt (XferCompl) Applies to IN and OUT endpoints. When Scatter/Gather DMA mode is enabled - For IN endpoint this field indicates that the requested data from the descriptor is moved from external system memory to internal FIFO. - For OUT endpoint this field indicates that the requested data from the internal FIFO is moved to external system memory. This interrupt is generated only when the corresponding endpoint descriptor is closed, and the IOC bit For the corresponding descriptor is Set. When Scatter/Gather DMA mode is disabled, this field indicates that the programmed transfer is complete on the AHB as well as on the USB, For this endpoint. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

dieptsiz8

Device IN Endpoint 8 Transfer Size Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00A10
i_usbotg_1_devgrp	0xFFB40800	0xFFB40A10

Offset: 0x210

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	mc RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
xfersize RW 0x0															

dieptsiz8 Fields

Bit	Name	Description	Access	Reset
30:29	mc	Applies to IN endpoints only. For periodic IN endpoints, this field indicates the number of packets that must be transmitted per microframe on the USB. The core uses this field to calculate the data PID for isochronous IN endpoints. 2'b01: 1 packet 2'b10: 2 packets 2'b11: 3 packets For non-periodic IN endpoints, this field is valid only in Internal DMA mode. It specifies the number of packets the core must fetchfor an IN endpoint before it switches to the endpoint pointed to by the Next Endpoint field of the Device Endpoint-n Control register (DIEPCTLn.NextEp) **Value** **Description** 0x1 PACKETONE 0x2 PACKETTWO 0x3 PACKETTHREE	RW	0x0

Bit	Name	Description	Access	Reset
28:19	pktcnt	Packet Count (PktCnt) Indicates the total number of USB packets that constitute the Transfer Size amount of data For endpoint 0. This field is decremented every time a packet (maximum size or short packet) is read from the TxFIFO.	RW	0x0
18:0	xfersize	Transfer Size (XferSize) Indicates the transfer size in bytes For endpoint 0. The core interrupts the application only after it has exhausted the transfer size amount of data. The transfer size can be Set to the maximum packet size of the endpoint, to be interrupted at the end of each packet. The core decrements this field every time a packet from the external memory is written to the TxFIFO.	RW	0x0

diepdma8

Device IN Endpoint 8 DMA Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00A14
i_usbotg_1_devgrp	0xFFB40800	0xFFB40A14

Offset: 0x214

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepdma8 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
diepdma8 RW 0x0															

diepdma8 Fields

Bit	Name	Description	Access	Reset
31:0	diepdma8	Holds the start address of the external memory for storing or fetching endpoint data. Note: For control endpoints, this field stores control OUT data packets as well as SETUP transaction data packets. When more than three SETUP packets are received back-to-back, the SETUP data packet in the memory is overwritten. This register is incremented on every AHB transaction. The application can give only a DWORD-aligned address. When Scatter/Gather DMA mode is not enabled, the application programs the start address value in this field. When Scatter/Gather DMA mode is enabled, this field indicates the base pointer for the descriptor list.	RW	0x0

dtxfsts8

Device IN Endpoint Transmit FIFO Status Register 8

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00A18
i_usbotg_1_devgrp	0xFFB40800	0xFFB40A18

Offset: 0x218

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ineptxfspcavail RO 0x2000															

dtxfsts8 Fields

Bit	Name	Description	Access	Reset
15:0	ineptxfspcavail	IN Endpoint TxFIFO Space Avail (INEPTxFSpcAvail) Indicates the amount of free space available in the Endpoint TxFIFO. Values are in terms of 32-bit words. 16'h0: Endpoint TxFIFO is full 16'h1: 1 word available 16'h2: 2 words available 16'hn: n words available (where 0 n 32,768) 16'h8000: 32,768 words available Others: Reserved	RO	0x2000

diepdmab8

Device IN Endpoint 8 Buffer Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00A1C
i_usbotg_1_devgrp	0xFFB40800	0xFFB40A1C

Offset: 0x21C

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepdmab8 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
diepdmab8 RO 0x0															

diepdmab8 Fields

Bit	Name	Description	Access	Reset
31:0	diepdmab8	Holds the current buffer address.This register is updated as and when the data transfer for the corresponding end point is in progress. This register is present only in Scatter/Gather DMA mode. Otherwise this field is reserved.	RO	0x0

diepctl9

Device Control IN Endpoint 9 Control Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00A20
i_usbotg_1_devgrp	0xFFB40800	0xFFB40A20

Offset: 0x220

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	txfnum RW 0x0				stall RW 0x0	Reserved	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
usbactep RW 0x0	Reserved				mps RW 0x0										

diepctl9 Fields

Bit	Name	Description	Access	Reset
31	epena	Endpoint Enable (EPEna) Applies to IN and OUT endpoints. When Scatter/Gather DMA mode is enabled, For IN endpoints this bit indicates that the descriptor structure and data buffer with data ready to transmit is setup. For OUT endpoint it indicates that the descriptor structure and data buffer to receive data is setup. When Scatter/Gather DMA mode is enabledsuch as for buffer-pointer based DMA mode: - For IN endpoints, this bit indicates that data is ready to be transmitted on the endpoint. - For OUT endpoints, this bit indicates that the application has allocated the memory to start receiving data from the USB. - The core clears this bit before setting any of the following interrupts on this endpoint: SETUP Phase Done Endpoint Disabled Transfer Completed Note: For control endpoints in DMA mode, this bit must be set to be able to transfer SETUP data packets in memory. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
30	epdis	Endpoint Disable (EPDis) Applies to IN and OUT endpoints. The application sets this bit to stop transmitting/receiving data on an endpoint, even before the transfer for that endpoint is complete. The application must wait for the Endpoint Disabled interrupt before treating the endpoint as disabled. The core clears this bit before setting the Endpoint Disabled interrupt. The application must set this bit only if Endpoint Enable is already set for this endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
29	setd1pid	Set DATA1 PID (SetD1PID) Applies to interrupt/bulk IN and OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA1. This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. Set Odd (micro)frame (SetOddFr) Applies to isochronous IN and OUT endpoints only. Writing to this field sets the Even/Odd (micro)frame (EO_FrNum) field to odd (micro)frame. This field is not applicable for Scatter/Gather DMA mode. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	WO	0x0

Bit	Name	Description	Access	Reset
28	setd0pid	Set DATA0 PID (SetD0PID) Applies to interrupt/bulk IN and OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA0. This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. 1'b0 WO In non-Scatter/Gather DMA mode: Set Even (micro)frame (SetEvenFr) Applies to isochronous IN and OUT endpoints only. Writing to this field sets the Even/Odd (micro)frame (EO_FrNum) field to even (micro) frame. When Scatter/Gather DMA mode is enabled, this field is reserved. The frame number in which to send data is in the transmit descriptor structure. The frame in which to receive data is updated in receive descriptor structure. **Value: Description** 0x0: DISABLED 0x1: ENABLED	WO	0x0
27	snak	Set NAK (SNAK) A write to this bit sets the NAK bit For the endpoint. Using this bit, the application can control the transmission of NAK handshakes on an endpoint. The core can also Set this bit For an endpoint after a SETUP packet is received on that endpoint. **Value: Description** 0x0: INACTIVE 0x1: ACTIVE	WO	0x0

Bit	Name	Description	Access	Reset
26	cnak	Clear NAK (CNAK) A write to this bit clears the NAK bit For the endpoint. Value: Description 0x0: INACTIVE 0x1: ACTIVE	WO	0x0
25:22	txfnum	TxFIFO Number (TxFNum) Shared FIFO Operation non-periodic endpoints must set this bit to zero. Periodic endpoints must map this to the corresponding Periodic TxFIFO number. 4'h0: Non-Periodic TxFIFO Others: Specified Periodic TxFIFO.number Note: An interrupt IN endpoint can be configured as a non-periodic endpoint for applications such as mass storage. The core treats an IN endpoint as a non-periodic endpoint if the TxFNum field is set to 0. Otherwise, a separate periodic FIFO must be allocated for an interrupt IN endpoint, and the number of this FIFO must be programmed into the TxFNum field. Configuring an interrupt IN endpoint as a non-periodic endpoint saves the extra periodic FIFO area. Dedicated FIFO Operationthese bits specify the FIFO number associated with this endpoint. Each active IN endpoint must be programmed to a separate FIFO number. This field is valid only for IN endpoints.	RW	0x0

Bit	Name	Description	Access	Reset
21	stall	STALL Handshake (Stall) Applies to non-control, non-isochronous IN and OUT endpoints only. The application sets this bit to stall all tokens from the USB host to this endpoint. If a NAK bit, Global Non-periodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority. Only the application can clear this bit, never the core. 1'b0 R_W Applies to control endpoints only. The application can only set this bit, and the core clears it, when a SETUP token is received for this endpoint. If a NAK bit, Global Non-periodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority. Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake. **Value: Description** 0x0: INACTIVE 0x1: ACTIVE	RW	0x0
19:18	eptype	Endpoint Type (EPType) This is the transfer type supported by this logical endpoint. 2'b00: Control 2'b01: Isochronous 2'b10: Bulk 2'b11: Interrupt **Value: Description** 0x0: CONTROL 0x1: ISOCHRONOUS 0x2: BULK 0x3: INTERRUP	RW	0x0

Bit	Name	Description	Access	Reset
17	naksts	NAK Status (NAKSts) Indicates the following: 1'b0: The core is transmitting non-NAK handshakes based on the FIFO status. 1'b1: The core is transmitting NAK handshakes on this endpoint. When either the application or the core sets this bit: The core stops receiving any data on an OUT endpoint, even if there is space in the RxFIFO to accommodate the incoming packet. For non-isochronous IN endpoints: The core stops transmitting any data on an IN endpoint, even if there data is available in the TxFIFO. For isochronous IN endpoints: The core sends out a zero-length data packet, even if there data is available in the TxFIFO. Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake. **Value** — **Description** 0x0 — NONNAK 0x1 — NAK	RO	0x0

Bit	Name	Description	Access	Reset
16	dpid	Endpoint Data PID (DPID) Applies to interrupt/bulk IN and OUT endpoints only. Contains the PID of the packet to be received or transmitted on this endpoint. The application must program the PID of the first packet to be received or transmitted on this endpoint, after the endpoint is activated. The applications use the SetD1PID and SetD0PID fields of this register to program either DATA0 or DATA1 PID. 1'b0: DATA0 1'b1: DATA1 This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. 1'b0 RO Even/Odd (Micro)Frame (EO_FrNum) In non-Scatter/Gather DMA mode: Applies to isochronous IN and OUT endpoints only. Indicates the (micro)frame number in which the core transmits/ receives isochronous data for this endpoint. The application must program the even/odd (micro) frame number in which it intends to transmit/receive isochronous data for this endpoint using the SetEvnFr and SetOddFr fields in this register. 1'b0: Even (micro)frame 1'b1: Odd (micro)frame When Scatter/Gather DMA mode is enabled, this field is reserved. The frame number in which to send data is provided in the transmit descriptor structure. The frame in which data is received is updated in receive descriptor structure. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RO	0x0

Bit	Name	Description	Access	Reset
15	usbactep	USB Active Endpoint (USBActEP) Indicates whether this endpoint is active in the current configuration and interface. The core clears this bit for all endpoints (other than EP 0) after detecting a USB reset. After receiving the SetConfiguration and SetInterface commands, the applica-tion must program endpoint registers accordingly and set this bit. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
10:0	mps	Maximum Packet Size (MPS) The application must program this field with the maximum packet size for the current logical endpoint. This value is in bytes.	RW	0x0

diepint9

Device IN Endpoint 9 Interrupt Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00A28
i_usbotg_1_devgrp	0xFFB40800	0xFFB40A28

Offset: 0x228

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	txfifoundrn RW 0x0	txfempp RO 0x1	inepnakeff RW 0x0	intknepmis RW 0x0	intkntxfempp RW 0x0	timeout RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0

diepint9 Fields

Bit	Name	Description	Access	Reset
14	nyetintrpt	NYET Interrupt (NYETIntrpt) The core generates this interrupt when a NYET response is transmitted for a non isochronous OUT endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
13	nakintrpt	NAK Interrupt (NAKInterrupt) The core generates this interrupt when a NAK is transmitted or received by the device. In case of isochronous IN endpoints the interrupt gets generated when a zero length packet is transmitted due to un-availability of data in the TXFifo. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
12	bbleerr	NAK Interrupt (BbleErr) The core generates this interrupt when babble is received for the endpoint. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
11	pktdrpsts	Packet Drop Status (PktDrpSts) This bit indicates to the application that an ISOC OUT packet has been dropped. This bit does not have an associated mask bit and does not generate an interrupt. Dependency: This bit is valid in non Scatter/Gather DMA mode when periodic transfer interrupt feature is selected. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
9	bnaintr	BNA (Buffer Not Available) Interrupt (BNAIntr) This bit is valid only when Scatter/Gather DMA mode is enabled. The core generates this interrupt when the descriptor accessed is not ready For the Core to process, such as Host busy or DMA done **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
8	txfifoundrn	Fifo Underrun (TxfifoUndrn) Applies to IN endpoints Only This bit is valid only If thresholding is enabled. The core generates this interrupt when it detects a transmit FIFO underrun condition For this endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
7	txfemp	Transmit FIFO Empty (TxFEmp) This bit is valid only For IN Endpoints This interrupt is asserted when the TxFIFO For this endpoint is either half or completely empty. The half or completely empty status is determined by the TxFIFO Empty Level bit in the Core AHB Configuration register (GAHBCFG.NPTxFEmpLvl)). **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RO	0x1

Bit	Name	Description	Access	Reset
6	inepnakeff	IN Endpoint NAK Effective (INEPNakEff) Applies to periodic IN endpoints only. This bit can be cleared when the application clears the IN endpoint NAK by writing to DIEPCTLn.CNAK. This interrupt indicates that the core has sampled the NAK bit Set (either by the application or by the core). The interrupt indicates that the IN endpoint NAK bit Set by the application has taken effect in the core. This interrupt does not guarantee that a NAK handshake is sent on the USB. A STALL bit takes priority over a NAK bit. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
5	intknepmis	IN Token Received with EP Mismatch (INTknEPMis) Applies to non-periodic IN endpoints only. Indicates that the data in the top of the non-periodic TxFIFO belongs to an endpoint other than the one For which the IN token was received. This interrupt is asserted on the endpoint For which the IN token was received. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
4	intkntxfemp	IN Token Received When TxFIFO is Empty (INTknTXFEmp) Applies to non-periodic IN endpoints only. Indicates that an IN token was received when the associated TxFIFO (periodic/non-periodic) was empty. This interrupt is asserted on the endpoint For which the IN token was received. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
3	timeout	Timeout Condition (TimeOUT) In shared TX FIFO mode, applies to non-isochronous IN endpoints only. In dedicated FIFO mode, applies only to Control IN endpoints. In Scatter/Gather DMA mode, the TimeOUT interrupt is not asserted. Indicates that the core has detected a timeout condition on the USB For the last IN token on this endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
2	ahberr	AHB Error (AHBErr) Applies to IN and OUT endpoints. This is generated only in Internal DMA mode when there is an AHB error during an AHB read/write. The application can read the corresponding endpoint DMA address register to get the error address. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
1	epdisbld	Endpoint Disabled Interrupt (EPDisbld) Applies to IN and OUT endpoints. This bit indicates that the endpoint is disabled per the application's request. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
0	xfercompl	Transfer Completed Interrupt (XferCompl) Applies to IN and OUT endpoints. When Scatter/Gather DMA mode is enabled - For IN endpoint this field indicates that the requested data from the descriptor is moved from external system memory to internal FIFO. - For OUT endpoint this field indicates that the requested data from the internal FIFO is moved to external system memory. This interrupt is generated only when the corresponding endpoint descriptor is closed, and the IOC bit For the corresponding descriptor is Set. When Scatter/Gather DMA mode is disabled, this field indicates that the programmed transfer is complete on the AHB as well as on the USB, For this endpoint. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

dieptsiz9

Device IN Endpoint 9 Transfer Size Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00A30
i_usbotg_1_devgrp	0xFFB40800	0xFFB40A30

Offset: 0x230

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	mc RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
xfersize RW 0x0															

dieptsiz9 Fields

Bit	Name	Description	Access	Reset
30:29	mc	Applies to IN endpoints only. For periodic IN endpoints, this field indicates the number of packets that must be transmitted per microframe on the USB. The core uses this field to calculate the data PID for isochronous IN endpoints. 2'b01: 1 packet 2'b10: 2 packets 2'b11: 3 packets For non-periodic IN endpoints, this field is valid only in Internal DMA mode. It specifies the number of packets the core must fetchfor an IN endpoint before it switches to the endpoint pointed to by the Next Endpoint field of the Device Endpoint-n Control register (DIEPCTLn.NextEp) **Value** — **Description** 0x1 — PACKETONE 0x2 — PACKETTWO 0x3 — PACKETTHREE	RW	0x0

Bit	Name	Description	Access	Reset
28:19	pktcnt	Packet Count (PktCnt) Indicates the total number of USB packets that constitute the Transfer Size amount of data For endpoint 0. This field is decremented every time a packet (maximum size or short packet) is read from the TxFIFO.	RW	0x0
18:0	xfersize	Transfer Size (XferSize) Indicates the transfer size in bytes For endpoint 0. The core interrupts the application only after it has exhausted the transfer size amount of data. The transfer size can be Set to the maximum packet size of the endpoint, to be interrupted at the end of each packet. The core decrements this field every time a packet from the external memory is written to the TxFIFO.	RW	0x0

diepdma9

Device IN Endpoint 9 DMA Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00A34
i_usbotg_1_devgrp	0xFFB40800	0xFFB40A34

Offset: 0x234

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepdma9 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
diepdma9 RW 0x0															

diepdma9 Fields

Bit	Name	Description	Access	Reset
31:0	diepdma9	Holds the start address of the external memory for storing or fetching endpoint data. Note: For control endpoints, this field stores control OUT data packets as well as SETUP transaction data packets. When more than three SETUP packets are received back-to-back, the SETUP data packet in the memory is overwritten. This register is incremented on every AHB transaction. The application can give only a DWORD-aligned address. When Scatter/Gather DMA mode is not enabled, the application programs the start address value in this field. When Scatter/Gather DMA mode is enabled, this field indicates the base pointer for the descriptor list.	RW	0x0

dtxfsts9

Device IN Endpoint Transmit FIFO Status Register 9

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00A38
i_usbotg_1_devgrp	0xFFB40800	0xFFB40A38

Offset: 0x238

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ineptxfspcavail RO 0x2000															

dtxfsts9 Fields

Bit	Name	Description	Access	Reset
15:0	ineptxfspcavail	IN Endpoint TxFIFO Space Avail (INEPTxFSpcAvail) Indicates the amount of free space available in the Endpoint TxFIFO. Values are in terms of 32-bit words. 16'h0: Endpoint TxFIFO is full 16'h1: 1 word available 16'h2: 2 words available 16'hn: n words available (where 0 n 32,768) 16'h8000: 32,768 words available Others: Reserved	RO	0x2000

diepdmab9

Device IN Endpoint 9 Buffer Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00A3C
i_usbotg_1_devgrp	0xFFB40800	0xFFB40A3C

Offset: 0x23C

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepdmab9 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
diepdmab9 RO 0x0															

diepdmab9 Fields

Bit	Name	Description	Access	Reset
31:0	diepdmab9	Holds the current buffer address.This register is updated as and when the data transfer for the corresponding end point is in progress. This register is present only in Scatter/Gather DMA mode. Otherwise this field is reserved.	RO	0x0

diepctl10

Device Control IN Endpoint 10 Control Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00A40
i_usbotg_1_devgrp	0xFFB40800	0xFFB40A40

Offset: 0x240

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	txfnum RW 0x0				stall RW 0x0	Reserved	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
usbactep RW 0x0	Reserved				mps RW 0x0										

diepctl10 Fields

Bit	Name	Description	Access	Reset
31	epena	Endpoint Enable (EPEna) Applies to IN and OUT endpoints. When Scatter/Gather DMA mode is enabled, For IN endpoints this bit indicates that the descriptor structure and data buffer with data ready to transmit is setup. For OUT endpoint it indicates that the descriptor structure and data buffer to receive data is setup. When Scatter/Gather DMA mode is enabledsuch as for buffer-pointer based DMA mode: - For IN endpoints, this bit indicates that data is ready to be transmitted on the endpoint. - For OUT endpoints, this bit indicates that the application has allocated the memory to start receiving data from the USB. - The core clears this bit before setting any of the following interrupts on this endpoint: SETUP Phase Done Endpoint Disabled Transfer Completed Note: For control endpoints in DMA mode, this bit must be set to be able to transfer SETUP data packets in memory. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
30	epdis	Endpoint Disable (EPDis) Applies to IN and OUT endpoints. The application sets this bit to stop transmitting/receiving data on an endpoint, even before the transfer for that endpoint is complete. The application must wait for the Endpoint Disabled interrupt before treating the endpoint as disabled. The core clears this bit before setting the Endpoint Disabled interrupt. The application must set this bit only if Endpoint Enable is already set for this endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
29	setd1pid	Set DATA1 PID (SetD1PID) Applies to interrupt/bulk IN and OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA1. This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. Set Odd (micro)frame (SetOddFr) Applies to isochronous IN and OUT endpoints only. Writing to this field sets the Even/Odd (micro)frame (EO_FrNum) field to odd (micro)frame. This field is not applicable for Scatter/Gather DMA mode. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	WO	0x0

Bit	Name	Description	Access	Reset
28	setd0pid	Set DATA0 PID (SetD0PID) Applies to interrupt/bulk IN and OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA0. This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. 1'b0 WO In non-Scatter/Gather DMA mode: Set Even (micro)frame (SetEvenFr) Applies to isochronous IN and OUT endpoints only. Writing to this field sets the Even/Odd (micro)frame (EO_FrNum) field to even (micro) frame. When Scatter/Gather DMA mode is enabled, this field is reserved. The frame number in which to send data is in the transmit descriptor structure. The frame in which to receive data is updated in receive descriptor structure. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	WO	0x0
27	snak	Set NAK (SNAK) A write to this bit sets the NAK bit For the endpoint. Using this bit, the application can control the transmission of NAK handshakes on an endpoint. The core can also Set this bit For an endpoint after a SETUP packet is received on that endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	WO	0x0

Bit	Name	Description	Access	Reset
26	cnak	Clear NAK (CNAK) A write to this bit clears the NAK bit For the endpoint. Value: 0x0 — INACTIVE Value: 0x1 — ACTIVE	WO	0x0
25:22	txfnum	TxFIFO Number (TxFNum) Shared FIFO Operation non-periodic endpoints must set this bit to zero. Periodic endpoints must map this to the corresponding Periodic TxFIFO number. 4'h0: Non-Periodic TxFIFO Others: Specified Periodic TxFIFO.number Note: An interrupt IN endpoint can be configured as a non-periodic endpoint for applications such as mass storage. The core treats an IN endpoint as a non-periodic endpoint if the TxFNum field is set to 0. Otherwise, a separate periodic FIFO must be allocated for an interrupt IN endpoint, and the number of this FIFO must be programmed into the TxFNum field. Configuring an interrupt IN endpoint as a non-periodic endpoint saves the extra periodic FIFO area. Dedicated FIFO Operationthese bits specify the FIFO number associated with this endpoint. Each active IN endpoint must be programmed to a separate FIFO number. This field is valid only for IN endpoints.	RW	0x0

Bit	Name	Description	Access	Reset
21	stall	STALL Handshake (Stall) Applies to non-control, non-isochronous IN and OUT endpoints only. The application sets this bit to stall all tokens from the USB host to this endpoint. If a NAK bit, Global Non-periodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority. Only the application can clear this bit, never the core. 1'b0 R_W Applies to control endpoints only. The application can only set this bit, and the core clears it, when a SETUP token is received for this endpoint. If a NAK bit, Global Non-periodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority. Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake. **Value: Description** 0x0: INACTIVE 0x1: ACTIVE	RW	0x0
19:18	eptype	Endpoint Type (EPType) This is the transfer type supported by this logical endpoint. 2'b00: Control 2'b01: Isochronous 2'b10: Bulk 2'b11: Interrupt **Value: Description** 0x0: CONTROL 0x1: ISOCHRONOUS 0x2: BULK 0x3: INTERRUP	RW	0x0

Bit	Name	Description	Access	Reset
17	naksts	NAK Status (NAKSts) Indicates the following: 1'b0: The core is transmitting non-NAK handshakes based on the FIFO status. 1'b1: The core is transmitting NAK handshakes on this endpoint. When either the application or the core sets this bit: The core stops receiving any data on an OUT endpoint, even if there is space in the RxFIFO to accommodate the incoming packet. For non-isochronous IN endpoints: The core stops transmitting any data on an IN endpoint, even if there data is available in the TxFIFO. For isochronous IN endpoints: The core sends out a zero-length data packet, even if there data is available in the TxFIFO. Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake. **Value** — **Description** 0x0 — NONNAK 0x1 — NAK	RO	0x0

Bit	Name	Description	Access	Reset
16	dpid	Endpoint Data PID (DPID) Applies to interrupt/bulk IN and OUT endpoints only. Contains the PID of the packet to be received or transmitted on this endpoint. The application must program the PID of the first packet to be received or transmitted on this endpoint, after the endpoint is activated. The applications use the SetD1PID and SetD0PID fields of this register to program either DATA0 or DATA1 PID. 1'b0: DATA0 1'b1: DATA1 This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. 1'b0 RO Even/Odd (Micro)Frame (EO_FrNum) In non-Scatter/Gather DMA mode: Applies to isochronous IN and OUT endpoints only. Indicates the (micro)frame number in which the core transmits/ receives isochronous data for this endpoint. The application must program the even/odd (micro) frame number in which it intends to transmit/receive isochronous data for this endpoint using the SetEvnFr and SetOddFr fields in this register. 1'b0: Even (micro)frame 1'b1: Odd (micro)frame When Scatter/Gather DMA mode is enabled, this field is reserved. The frame number in which to send data is provided in the transmit descriptor structure. The frame in which data is received is updated in receive descriptor structure. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RO	0x0

Bit	Name	Description	Access	Reset
15	usbactep	USB Active Endpoint (USBActEP) Indicates whether this endpoint is active in the current configuration and interface. The core clears this bit for all endpoints (other than EP 0) after detecting a USB reset. After receiving the SetConfiguration and SetInterface commands, the applica-tion must program endpoint registers accordingly and set this bit. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
10:0	mps	Maximum Packet Size (MPS) The application must program this field with the maximum packet size for the current logical endpoint. This value is in bytes.	RW	0x0

diepint10

Device IN Endpoint 10 Interrupt Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00A48
i_usbotg_1_devgrp	0xFFB40800	0xFFB40A48

Offset: 0x248

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	txfifoundrn RW 0x0	txfemp RO 0x1	inepnakeff RW 0x0	intknepmis RW 0x0	intkntxfemp RW 0x0	timeout RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0

diepint10 Fields

Bit	Name	Description	Access	Reset
14	nyetintrpt	NYET Interrupt (NYETIntrpt) The core generates this interrupt when a NYET response is transmitted for a non isochronous OUT endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
13	nakintrpt	NAK Interrupt (NAKInterrupt) The core generates this interrupt when a NAK is transmitted or received by the device. In case of isochronous IN endpoints the interrupt gets generated when a zero length packet is transmitted due to un-availability of data in the TXFifo. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
12	bbleerr	NAK Interrupt (BbleErr) The core generates this interrupt when babble is received for the endpoint. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
11	pktdrpsts	Packet Drop Status (PktDrpSts) This bit indicates to the application that an ISOC OUT packet has been dropped. This bit does not have an associated mask bit and does not generate an interrupt. Dependency: This bit is valid in non Scatter/Gather DMA mode when periodic transfer interrupt feature is selected. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
9	bnaintr	BNA (Buffer Not Available) Interrupt (BNAIntr) This bit is valid only when Scatter/Gather DMA mode is enabled. The core generates this interrupt when the descriptor accessed is not ready For the Core to process, such as Host busy or DMA done **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
8	txfifoundrn	Fifo Underrun (TxfifoUndrn) Applies to IN endpoints Only This bit is valid only If thresholding is enabled. The core generates this interrupt when it detects a transmit FIFO underrun condition For this endpoint. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
7	txfemp	Transmit FIFO Empty (TxFEmp) This bit is valid only For IN Endpoints This interrupt is asserted when the TxFIFO For this endpoint is either half or completely empty. The half or completely empty status is determined by the TxFIFO Empty Level bit in the Core AHB Configuration register (GAHBCFG.NPTxFEmpLvl)). **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x1

Bit	Name	Description	Access	Reset
6	inepnakeff	IN Endpoint NAK Effective (INEPNakEff) Applies to periodic IN endpoints only. This bit can be cleared when the application clears the IN endpoint NAK by writing to DIEPCTLn.CNAK. This interrupt indicates that the core has sampled the NAK bit Set (either by the application or by the core). The interrupt indicates that the IN endpoint NAK bit Set by the application has taken effect in the core. This interrupt does not guarantee that a NAK handshake is sent on the USB. A STALL bit takes priority over a NAK bit. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
5	intknepmis	IN Token Received with EP Mismatch (INTknEPMis) Applies to non-periodic IN endpoints only. Indicates that the data in the top of the non-periodic TxFIFO belongs to an endpoint other than the one For which the IN token was received. This interrupt is asserted on the endpoint For which the IN token was received. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
4	intkntxfemp	IN Token Received When TxFIFO is Empty (INTknTXFEmp) Applies to non-periodic IN endpoints only. Indicates that an IN token was received when the associated TxFIFO (periodic/non-periodic) was empty. This interrupt is asserted on the endpoint For which the IN token was received. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
3	timeout	Timeout Condition (TimeOUT) In shared TX FIFO mode, applies to non-isochronous IN endpoints only. In dedicated FIFO mode, applies only to Control IN endpoints. In Scatter/Gather DMA mode, the TimeOUT interrupt is not asserted. Indicates that the core has detected a timeout condition on the USB For the last IN token on this endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
2	ahberr	AHB Error (AHBErr) Applies to IN and OUT endpoints. This is generated only in Internal DMA mode when there is an AHB error during an AHB read/write. The application can read the corresponding endpoint DMA address register to get the error address. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
1	epdisbld	Endpoint Disabled Interrupt (EPDisbld) Applies to IN and OUT endpoints. This bit indicates that the endpoint is disabled per the application's request. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
0	xfercompl	Transfer Completed Interrupt (XferCompl) Applies to IN and OUT endpoints. When Scatter/Gather DMA mode is enabled - For IN endpoint this field indicates that the requested data from the descriptor is moved from external system memory to internal FIFO. - For OUT endpoint this field indicates that the requested data from the internal FIFO is moved to external system memory. This interrupt is generated only when the corresponding endpoint descriptor is closed, and the IOC bit For the corresponding descriptor is Set. When Scatter/Gather DMA mode is disabled, this field indicates that the programmed transfer is complete on the AHB as well as on the USB, For this endpoint. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

dieptsiz10

Device IN Endpoint 10 Transfer Size Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00A50
i_usbotg_1_devgrp	0xFFB40800	0xFFB40A50

Offset: 0x250

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	mc RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
xfersize RW 0x0															

dieptsiz10 Fields

Bit	Name	Description	Access	Reset
30:29	mc	Applies to IN endpoints only. For periodic IN endpoints, this field indicates the number of packets that must be transmitted per microframe on the USB. The core uses this field to calculate the data PID for isochronous IN endpoints. 2'b01: 1 packet 2'b10: 2 packets 2'b11: 3 packets For non-periodic IN endpoints, this field is valid only in Internal DMA mode. It specifies the number of packets the core must fetchfor an IN endpoint before it switches to the endpoint pointed to by the Next Endpoint field of the Device Endpoint-n Control register (DIEPCTLn.NextEp) **Value** — **Description** 0x1 — PACKETONE 0x2 — PACKETTWO 0x3 — PACKETTHREE	RW	0x0

Bit	Name	Description	Access	Reset
28:19	pktcnt	Packet Count (PktCnt) Indicates the total number of USB packets that constitute the Transfer Size amount of data For endpoint 0. This field is decremented every time a packet (maximum size or short packet) is read from the TxFIFO.	RW	0x0
18:0	xfersize	Transfer Size (XferSize) Indicates the transfer size in bytes For endpoint 0. The core interrupts the application only after it has exhausted the transfer size amount of data. The transfer size can be Set to the maximum packet size of the endpoint, to be interrupted at the end of each packet. The core decrements this field every time a packet from the external memory is written to the TxFIFO.	RW	0x0

diepdma10

Device IN Endpoint 10 DMA Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00A54
i_usbotg_1_devgrp	0xFFB40800	0xFFB40A54

Offset: 0x254

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepdma10 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
diepdma10 RW 0x0															

diepdma10 Fields

Bit	Name	Description	Access	Reset
31:0	diepdma10	Holds the start address of the external memory for storing or fetching endpoint data. Note: For control endpoints, this field stores control OUT data packets as well as SETUP transaction data packets. When more than three SETUP packets are received back-to-back, the SETUP data packet in the memory is overwritten. This register is incremented on every AHB transaction. The application can give only a DWORD-aligned address. When Scatter/Gather DMA mode is not enabled, the application programs the start address value in this field. When Scatter/Gather DMA mode is enabled, this field indicates the base pointer for the descriptor list.	RW	0x0

dtxfsts10

Device IN Endpoint Transmit FIFO Status Register 10

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00A58
i_usbotg_1_devgrp	0xFFB40800	0xFFB40A58

Offset: 0x258

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ineptxfspcavail RO 0x2000															

dtxfsts10 Fields

Bit	Name	Description	Access	Reset
15:0	ineptxfspcavail	IN Endpoint TxFIFO Space Avail (INEPTxFSpcAvail) Indicates the amount of free space available in the Endpoint TxFIFO. Values are in terms of 32-bit words. 16'h0: Endpoint TxFIFO is full 16'h1: 1 word available 16'h2: 2 words available 16'hn: n words available (where 0 n 32,768) 16'h8000: 32,768 words available Others: Reserved	RO	0x2000

diepdmab10

Device IN Endpoint 10 Buffer Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00A5C
i_usbotg_1_devgrp	0xFFB40800	0xFFB40A5C

Offset: 0x25C

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepdmab10 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
diepdmab10 RO 0x0															

diepdmab10 Fields

Bit	Name	Description	Access	Reset
31:0	diepdmab10	Holds the current buffer address.This register is updated as and when the data transfer for the corresponding end point is in progress. This register is present only in Scatter/Gather DMA mode. Otherwise this field is reserved.	RO	0x0

diepctl11

Device Control IN Endpoint 11 Control Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00A60
i_usbotg_1_devgrp	0xFFB40800	0xFFB40A60

Offset: 0x260

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	txfnum RW 0x0				stall RW 0x0	Reserved	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
usbactep RW 0x0	Reserved				mps RW 0x0										

diepctl11 Fields

Bit	Name	Description	Access	Reset
31	epena	Endpoint Enable (EPEna) Applies to IN and OUT endpoints. When Scatter/Gather DMA mode is enabled, For IN endpoints this bit indicates that the descriptor structure and data buffer with data ready to transmit is setup. For OUT endpoint it indicates that the descriptor structure and data buffer to receive data is setup. When Scatter/Gather DMA mode is enabledsuch as for buffer-pointer based DMA mode: - For IN endpoints, this bit indicates that data is ready to be transmitted on the endpoint. - For OUT endpoints, this bit indicates that the application has allocated the memory to start receiving data from the USB. - The core clears this bit before setting any of the following interrupts on this endpoint: SETUP Phase Done Endpoint Disabled Transfer Completed Note: For control endpoints in DMA mode, this bit must be set to be able to transfer SETUP data packets in memory. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
30	epdis	Endpoint Disable (EPDis) Applies to IN and OUT endpoints. The application sets this bit to stop transmitting/receiving data on an endpoint, even before the transfer for that endpoint is complete. The application must wait for the Endpoint Disabled interrupt before treating the endpoint as disabled. The core clears this bit before setting the Endpoint Disabled interrupt. The application must set this bit only if Endpoint Enable is already set for this endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
29	setd1pid	Set DATA1 PID (SetD1PID) Applies to interrupt/bulk IN and OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA1. This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. Set Odd (micro)frame (SetOddFr) Applies to isochronous IN and OUT endpoints only. Writing to this field sets the Even/Odd (micro)frame (EO_FrNum) field to odd (micro)frame. This field is not applicable for Scatter/Gather DMA mode. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	WO	0x0

Bit	Name	Description	Access	Reset
28	setd0pid	Set DATA0 PID (SetD0PID) Applies to interrupt/bulk IN and OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA0. This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. 1'b0 WO In non-Scatter/Gather DMA mode: Set Even (micro)frame (SetEvenFr) Applies to isochronous IN and OUT endpoints only. Writing to this field sets the Even/Odd (micro)frame (EO_FrNum) field to even (micro) frame. When Scatter/Gather DMA mode is enabled, this field is reserved. The frame number in which to send data is in the transmit descriptor structure. The frame in which to receive data is updated in receive descriptor structure. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	WO	0x0
27	snak	Set NAK (SNAK) A write to this bit sets the NAK bit For the endpoint. Using this bit, the application can control the transmission of NAK handshakes on an endpoint. The core can also Set this bit For an endpoint after a SETUP packet is received on that endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	WO	0x0

Bit	Name	Description	Access	Reset
26	cnak	Clear NAK (CNAK) A write to this bit clears the NAK bit For the endpoint. Value: Description 0x0: INACTIVE 0x1: ACTIVE	WO	0x0
25:22	txfnum	TxFIFO Number (TxFNum) Shared FIFO Operation non-periodic endpoints must set this bit to zero. Periodic endpoints must map this to the corresponding Periodic TxFIFO number. 4'h0: Non-Periodic TxFIFO Others: Specified Periodic TxFIFO.number Note: An interrupt IN endpoint can be configured as a non-periodic endpoint for applications such as mass storage. The core treats an IN endpoint as a non-periodic endpoint if the TxFNum field is set to 0. Otherwise, a separate periodic FIFO must be allocated for an interrupt IN endpoint, and the number of this FIFO must be programmed into the TxFNum field. Configuring an interrupt IN endpoint as a non-periodic endpoint saves the extra periodic FIFO area. Dedicated FIFO Operationthese bits specify the FIFO number associated with this endpoint. Each active IN endpoint must be programmed to a separate FIFO number. This field is valid only for IN endpoints.	RW	0x0

Bit	Name	Description	Access	Reset
21	stall	STALL Handshake (Stall) Applies to non-control, non-isochronous IN and OUT endpoints only. The application sets this bit to stall all tokens from the USB host to this endpoint. If a NAK bit, Global Non-periodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority. Only the application can clear this bit, never the core. 1'b0 R_W Applies to control endpoints only. The application can only set this bit, and the core clears it, when a SETUP token is received for this endpoint. If a NAK bit, Global Non-periodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority. Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake. **Value: Description** 0x0: INACTIVE 0x1: ACTIVE	RW	0x0
19:18	eptype	Endpoint Type (EPType) This is the transfer type supported by this logical endpoint. 2'b00: Control 2'b01: Isochronous 2'b10: Bulk 2'b11: Interrupt **Value: Description** 0x0: CONTROL 0x1: ISOCHRONOUS 0x2: BULK 0x3: INTERRUP	RW	0x0

Bit	Name	Description	Access	Reset
17	naksts	NAK Status (NAKSts) Indicates the following: 1'b0: The core is transmitting non-NAK handshakes based on the FIFO status. 1'b1: The core is transmitting NAK handshakes on this endpoint. When either the application or the core sets this bit: The core stops receiving any data on an OUT endpoint, even if there is space in the RxFIFO to accommodate the incoming packet. For non-isochronous IN endpoints: The core stops transmitting any data on an IN endpoint, even if there data is available in the TxFIFO. For isochronous IN endpoints: The core sends out a zero-length data packet, even if there data is available in the TxFIFO. Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake. **Value** — **Description** 0x0 — NONNAK 0x1 — NAK	RO	0x0

Bit	Name	Description	Access	Reset
16	dpid	Endpoint Data PID (DPID) Applies to interrupt/bulk IN and OUT endpoints only. Contains the PID of the packet to be received or transmitted on this endpoint. The application must program the PID of the first packet to be received or transmitted on this endpoint, after the endpoint is activated. The applications use the SetD1PID and SetD0PID fields of this register to program either DATA0 or DATA1 PID. 1'b0: DATA0 1'b1: DATA1 This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. 1'b0 RO Even/Odd (Micro)Frame (EO_FrNum) In non-Scatter/Gather DMA mode: Applies to isochronous IN and OUT endpoints only. Indicates the (micro)frame number in which the core transmits/receives isochronous data for this endpoint. The application must program the even/odd (micro) frame number in which it intends to transmit/receive isochronous data for this endpoint using the SetEvnFr and SetOddFr fields in this register. 1'b0: Even (micro)frame 1'b1: Odd (micro)frame When Scatter/Gather DMA mode is enabled, this field is reserved. The frame number in which to send data is provided in the transmit descriptor structure. The frame in which data is received is updated in receive descriptor structure. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RO	0x0

Bit	Name	Description	Access	Reset
15	usbactep	USB Active Endpoint (USBActEP) Indicates whether this endpoint is active in the current configuration and interface. The core clears this bit for all endpoints (other than EP 0) after detecting a USB reset. After receiving the SetConfiguration and SetInterface commands, the applica-tion must program endpoint registers accordingly and set this bit. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
10:0	mps	Maximum Packet Size (MPS) The application must program this field with the maximum packet size for the current logical endpoint. This value is in bytes.	RW	0x0

diepint11

Device IN Endpoint 11 Interrupt Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00A68
i_usbotg_1_devgrp	0xFFB40800	0xFFB40A68

Offset: 0x268

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	txfifoundrn RW 0x0	txfempp RO 0x1	inepnakeff RW 0x0	intknepmis RW 0x0	intkntxfempp RW 0x0	timeout RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0

diepint11 Fields

Bit	Name	Description	Access	Reset
14	nyetintrpt	NYET Interrupt (NYETIntrpt) The core generates this interrupt when a NYET response is transmitted for a non isochronous OUT endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
13	nakintrpt	NAK Interrupt (NAKInterrupt) The core generates this interrupt when a NAK is transmitted or received by the device. In case of isochronous IN endpoints the interrupt gets generated when a zero length packet is transmitted due to un-availability of data in the TXFifo. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
12	bbleerr	NAK Interrupt (BbleErr) The core generates this interrupt when babble is received for the endpoint. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
11	pktdrpsts	Packet Drop Status (PktDrpSts) This bit indicates to the application that an ISOC OUT packet has been dropped. This bit does not have an associated mask bit and does not generate an interrupt. Dependency: This bit is valid in non Scatter/Gather DMA mode when periodic transfer interrupt feature is selected. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
9	bnaintr	BNA (Buffer Not Available) Interrupt (BNAIntr) This bit is valid only when Scatter/Gather DMA mode is enabled. The core generates this interrupt when the descriptor accessed is not ready For the Core to process, such as Host busy or DMA done **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
8	txfifoundrn	Fifo Underrun (TxfifoUndrn) Applies to IN endpoints Only This bit is valid only If thresholding is enabled. The core generates this interrupt when it detects a transmit FIFO underrun condition For this endpoint. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
7	txfemp	Transmit FIFO Empty (TxFEmp) This bit is valid only For IN Endpoints This interrupt is asserted when the TxFIFO For this endpoint is either half or completely empty. The half or completely empty status is determined by the TxFIFO Empty Level bit in the Core AHB Configuration register (GAHBCFG.NPTxFEmpLvl)). **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x1

Bit	Name	Description	Access	Reset
6	inepnakeff	IN Endpoint NAK Effective (INEPNakEff) Applies to periodic IN endpoints only. This bit can be cleared when the application clears the IN endpoint NAK by writing to DIEPCTLn.CNAK. This interrupt indicates that the core has sampled the NAK bit Set (either by the application or by the core). The interrupt indicates that the IN endpoint NAK bit Set by the application has taken effect in the core. This interrupt does not guarantee that a NAK handshake is sent on the USB. A STALL bit takes priority over a NAK bit. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
5	intknepmis	IN Token Received with EP Mismatch (INTknEPMis) Applies to non-periodic IN endpoints only. Indicates that the data in the top of the non-periodic TxFIFO belongs to an endpoint other than the one For which the IN token was received. This interrupt is asserted on the endpoint For which the IN token was received. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
4	intkntxfemp	IN Token Received When TxFIFO is Empty (INTknTXFEmp) Applies to non-periodic IN endpoints only. Indicates that an IN token was received when the associated TxFIFO (periodic/non-periodic) was empty. This interrupt is asserted on the endpoint For which the IN token was received. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
3	timeout	Timeout Condition (TimeOUT) In shared TX FIFO mode, applies to non-isochronous IN endpoints only. In dedicated FIFO mode, applies only to Control IN endpoints. In Scatter/Gather DMA mode, the TimeOUT interrupt is not asserted. Indicates that the core has detected a timeout condition on the USB For the last IN token on this endpoint. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
2	ahberr	AHB Error (AHBErr) Applies to IN and OUT endpoints. This is generated only in Internal DMA mode when there is an AHB error during an AHB read/write. The application can read the corresponding endpoint DMA address register to get the error address. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
1	epdisbld	Endpoint Disabled Interrupt (EPDisbld) Applies to IN and OUT endpoints. This bit indicates that the endpoint is disabled per the application's request. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
0	xfercompl	Transfer Completed Interrupt (XferCompl) Applies to IN and OUT endpoints. When Scatter/Gather DMA mode is enabled - For IN endpoint this field indicates that the requested data from the descriptor is moved from external system memory to internal FIFO. - For OUT endpoint this field indicates that the requested data from the internal FIFO is moved to external system memory. This interrupt is generated only when the corresponding endpoint descriptor is closed, and the IOC bit For the corresponding descriptor is Set. When Scatter/Gather DMA mode is disabled, this field indicates that the programmed transfer is complete on the AHB as well as on the USB, For this endpoint. Value: 0x0 — INACTIVE Value: 0x1 — ACTIVE	RW	0x0

dieptsiz11

Device IN Endpoint 11 Transfer Size Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00A70
i_usbotg_1_devgrp	0xFFB40800	0xFFB40A70

Offset: 0x270

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	mc RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
xfersize RW 0x0															

dieptsiz11 Fields

Bit	Name	Description	Access	Reset
30:29	mc	Applies to IN endpoints only. For periodic IN endpoints, this field indicates the number of packets that must be transmitted per microframe on the USB. The core uses this field to calculate the data PID for isochronous IN endpoints. 2'b01: 1 packet 2'b10: 2 packets 2'b11: 3 packets For non-periodic IN endpoints, this field is valid only in Internal DMA mode. It specifies the number of packets the core must fetchfor an IN endpoint before it switches to the endpoint pointed to by the Next Endpoint field of the Device Endpoint-n Control register (DIEPCTLn.NextEp) **Value** **Description** 0x1 PACKETONE 0x2 PACKETTWO 0x3 PACKETTHREE	RW	0x0

Bit	Name	Description	Access	Reset
28:19	pktcnt	Packet Count (PktCnt) Indicates the total number of USB packets that constitute the Transfer Size amount of data For endpoint 0. This field is decremented every time a packet (maximum size or short packet) is read from the TxFIFO.	RW	0x0
18:0	xfersize	Transfer Size (XferSize) Indicates the transfer size in bytes For endpoint 0. The core interrupts the application only after it has exhausted the transfer size amount of data. The transfer size can be Set to the maximum packet size of the endpoint, to be interrupted at the end of each packet. The core decrements this field every time a packet from the external memory is written to the TxFIFO.	RW	0x0

diepdma11

Device IN Endpoint 11 DMA Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00A74
i_usbotg_1_devgrp	0xFFB40800	0xFFB40A74

Offset: 0x274

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepdma11 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
diepdma11 RW 0x0															

diepdma11 Fields

Bit	Name	Description	Access	Reset
31:0	diepdma11	Holds the start address of the external memory for storing or fetching endpoint data. Note: For control endpoints, this field stores control OUT data packets as well as SETUP transaction data packets. When more than three SETUP packets are received back-to-back, the SETUP data packet in the memory is overwritten. This register is incremented on every AHB transaction. The application can give only a DWORD-aligned address. When Scatter/Gather DMA mode is not enabled, the application programs the start address value in this field. When Scatter/Gather DMA mode is enabled, this field indicates the base pointer for the descriptor list.	RW	0x0

dtxfsts11

Device IN Endpoint Transmit FIFO Status Register 11

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00A78
i_usbotg_1_devgrp	0xFFB40800	0xFFB40A78

Offset: 0x278

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ineptxfspcavail RO 0x2000															

dtxfsts11 Fields

Bit	Name	Description	Access	Reset
15:0	ineptxfspcavail	IN Endpoint TxFIFO Space Avail (INEPTxFSpcAvail) Indicates the amount of free space available in the Endpoint TxFIFO. Values are in terms of 32-bit words. 16'h0: Endpoint TxFIFO is full 16'h1: 1 word available 16'h2: 2 words available 16'hn: n words available (where 0 n 32,768) 16'h8000: 32,768 words available Others: Reserved	RO	0x2000

diepdmab11

Device IN Endpoint 11 Buffer Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00A7C
i_usbotg_1_devgrp	0xFFB40800	0xFFB40A7C

Offset: 0x27C

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepdmab11 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
diepdmab11 RO 0x0															

diepdmab11 Fields

Bit	Name	Description	Access	Reset
31:0	diepdmab11	Holds the current buffer address.This register is updated as and when the data transfer for the corresponding end point is in progress. This register is present only in Scatter/Gather DMA mode. Otherwise this field is reserved.	RO	0x0

diepctl12

Device Control IN Endpoint 12 Control Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00A80
i_usbotg_1_devgrp	0xFFB40800	0xFFB40A80

Offset: 0x280

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	txfnum RW 0x0				stall RW 0x0	Reserved	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
usbactep RW 0x0	Reserved				mps RW 0x0										

diepctl12 Fields

Bit	Name	Description	Access	Reset
31	epena	Endpoint Enable (EPEna) Applies to IN and OUT endpoints. When Scatter/Gather DMA mode is enabled, For IN endpoints this bit indicates that the descriptor structure and data buffer with data ready to transmit is setup. For OUT endpoint it indicates that the descriptor structure and data buffer to receive data is setup. When Scatter/Gather DMA mode is enabledsuch as for buffer-pointer based DMA mode: - For IN endpoints, this bit indicates that data is ready to be transmitted on the endpoint. - For OUT endpoints, this bit indicates that the application has allocated the memory to start receiving data from the USB. - The core clears this bit before setting any of the following interrupts on this endpoint: SETUP Phase Done Endpoint Disabled Transfer Completed Note: For control endpoints in DMA mode, this bit must be set to be able to transfer SETUP data packets in memory. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
30	epdis	Endpoint Disable (EPDis) Applies to IN and OUT endpoints. The application sets this bit to stop transmitting/receiving data on an endpoint, even before the transfer for that endpoint is complete. The application must wait for the Endpoint Disabled interrupt before treating the endpoint as disabled. The core clears this bit before setting the Endpoint Disabled interrupt. The application must set this bit only if Endpoint Enable is already set for this endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
29	setd1pid	Set DATA1 PID (SetD1PID) Applies to interrupt/bulk IN and OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA1. This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. Set Odd (micro)frame (SetOddFr) Applies to isochronous IN and OUT endpoints only. Writing to this field sets the Even/Odd (micro)frame (EO_FrNum) field to odd (micro)frame. This field is not applicable for Scatter/Gather DMA mode. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	WO	0x0

Bit	Name	Description	Access	Reset
28	setd0pid	Set DATA0 PID (SetD0PID) Applies to interrupt/bulk IN and OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA0. This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. 1'b0 WO In non-Scatter/Gather DMA mode: Set Even (micro)frame (SetEvenFr) Applies to isochronous IN and OUT endpoints only. Writing to this field sets the Even/Odd (micro)frame (EO_FrNum) field to even (micro) frame. When Scatter/Gather DMA mode is enabled, this field is reserved. The frame number in which to send data is in the transmit descriptor structure. The frame in which to receive data is updated in receive descriptor structure. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	WO	0x0
27	snak	Set NAK (SNAK) A write to this bit sets the NAK bit For the endpoint. Using this bit, the application can control the transmission of NAK handshakes on an endpoint. The core can also Set this bit For an endpoint after a SETUP packet is received on that endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	WO	0x0

Bit	Name	Description	Access	Reset
26	cnak	Clear NAK (CNAK) A write to this bit clears the NAK bit For the endpoint. Value: 0x0 — Description: INACTIVE Value: 0x1 — Description: ACTIVE	WO	0x0
25:22	txfnum	TxFIFO Number (TxFNum) Shared FIFO Operation non-periodic endpoints must set this bit to zero. Periodic endpoints must map this to the corresponding Periodic TxFIFO number. 4'h0: Non-Periodic TxFIFO Others: Specified Periodic TxFIFO.number Note: An interrupt IN endpoint can be configured as a non-periodic endpoint for applications such as mass storage. The core treats an IN endpoint as a non-periodic endpoint if the TxFNum field is set to 0. Otherwise, a separate periodic FIFO must be allocated for an interrupt IN endpoint, and the number of this FIFO must be programmed into the TxFNum field. Configuring an interrupt IN endpoint as a non-periodic endpoint saves the extra periodic FIFO area. Dedicated FIFO Operationthese bits specify the FIFO number associated with this endpoint. Each active IN endpoint must be programmed to a separate FIFO number. This field is valid only for IN endpoints.	RW	0x0

Bit	Name	Description	Access	Reset
21	stall	STALL Handshake (Stall) Applies to non-control, non-isochronous IN and OUT endpoints only. The application sets this bit to stall all tokens from the USB host to this endpoint. If a NAK bit, Global Non-periodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority. Only the application can clear this bit, never the core. 1'b0 R_W Applies to control endpoints only. The application can only set this bit, and the core clears it, when a SETUP token is received for this endpoint. If a NAK bit, Global Non-periodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority. Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
19:18	eptype	Endpoint Type (EPType) This is the transfer type supported by this logical endpoint. 2'b00: Control 2'b01: Isochronous 2'b10: Bulk 2'b11: Interrupt **Value** **Description** 0x0 CONTROL 0x1 ISOCHRONOUS 0x2 BULK 0x3 INTERRUP	RW	0x0

Bit	Name	Description	Access	Reset
17	naksts	NAK Status (NAKSts) Indicates the following: 1'b0: The core is transmitting non-NAK handshakes based on the FIFO status. 1'b1: The core is transmitting NAK handshakes on this endpoint. When either the application or the core sets this bit: The core stops receiving any data on an OUT endpoint, even if there is space in the RxFIFO to accommodate the incoming packet. For non-isochronous IN endpoints: The core stops transmitting any data on an IN endpoint, even if there data is available in the TxFIFO. For isochronous IN endpoints: The core sends out a zero-length data packet, even if there data is available in the TxFIFO. Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake. **Value** — **Description** 0x0 — NONNAK 0x1 — NAK	RO	0x0

Bit	Name	Description	Access	Reset
16	dpid	Endpoint Data PID (DPID) Applies to interrupt/bulk IN and OUT endpoints only. Contains the PID of the packet to be received or transmitted on this endpoint. The application must program the PID of the first packet to be received or transmitted on this endpoint, after the endpoint is activated. The applications use the SetD1PID and SetD0PID fields of this register to program either DATA0 or DATA1 PID. 1'b0: DATA0 1'b1: DATA1 This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. 1'b0 RO Even/Odd (Micro)Frame (EO_FrNum) In non-Scatter/Gather DMA mode: Applies to isochronous IN and OUT endpoints only. Indicates the (micro)frame number in which the core transmits/receives isochronous data for this endpoint. The application must program the even/odd (micro) frame number in which it intends to transmit/receive isochronous data for this endpoint using the SetEvnFr and SetOddFr fields in this register. 1'b0: Even (micro)frame 1'b1: Odd (micro)frame When Scatter/Gather DMA mode is enabled, this field is reserved. The frame number in which to send data is provided in the transmit descriptor structure. The frame in which data is received is updated in receive descriptor structure. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RO	0x0

Bit	Name	Description	Access	Reset
15	usbactep	USB Active Endpoint (USBActEP) Indicates whether this endpoint is active in the current configuration and interface. The core clears this bit for all endpoints (other than EP 0) after detecting a USB reset. After receiving the SetConfiguration and SetInterface commands, the applica-tion must program endpoint registers accordingly and set this bit. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
10:0	mps	Maximum Packet Size (MPS) The application must program this field with the maximum packet size for the current logical endpoint. This value is in bytes.	RW	0x0

diepint12

Device IN Endpoint 12 Interrupt Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00A88
i_usbotg_1_devgrp	0xFFB40800	0xFFB40A88

Offset: 0x288

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	txfifoundrn RW 0x0	txfempp RO 0x1	inepnakeff RW 0x0	intknepmis RW 0x0	intkntxfempp RW 0x0	timeout RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0

diepint12 Fields

Bit	Name	Description	Access	Reset
14	nyetintrpt	NYET Interrupt (NYETIntrpt) The core generates this interrupt when a NYET response is transmitted for a non isochronous OUT endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
13	nakintrpt	NAK Interrupt (NAKInterrupt) The core generates this interrupt when a NAK is transmitted or received by the device. In case of isochronous IN endpoints the interrupt gets generated when a zero length packet is transmitted due to un-availability of data in the TXFifo. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
12	bbleerr	NAK Interrupt (BbleErr) The core generates this interrupt when babble is received for the endpoint. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
11	pktdrpsts	Packet Drop Status (PktDrpSts) This bit indicates to the application that an ISOC OUT packet has been dropped. This bit does not have an associated mask bit and does not generate an interrupt. Dependency: This bit is valid in non Scatter/Gather DMA mode when periodic transfer interrupt feature is selected. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
9	bnaintr	BNA (Buffer Not Available) Interrupt (BNAIntr) This bit is valid only when Scatter/Gather DMA mode is enabled. The core generates this interrupt when the descriptor accessed is not ready For the Core to process, such as Host busy or DMA done **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
8	txfifoundrn	Fifo Underrun (TxfifoUndrn) Applies to IN endpoints Only This bit is valid only If thresholding is enabled. The core generates this interrupt when it detects a transmit FIFO underrun condition For this endpoint. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
7	txfemp	Transmit FIFO Empty (TxFEmp) This bit is valid only For IN Endpoints This interrupt is asserted when the TxFIFO For this endpoint is either half or completely empty. The half or completely empty status is determined by the TxFIFO Empty Level bit in the Core AHB Configuration register (GAHBCFG.NPTxFEmpLvl)). **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x1

Bit	Name	Description	Access	Reset
6	inepnakeff	IN Endpoint NAK Effective (INEPNakEff) Applies to periodic IN endpoints only. This bit can be cleared when the application clears the IN endpoint NAK by writing to DIEPCTLn.CNAK. This interrupt indicates that the core has sampled the NAK bit Set (either by the application or by the core). The interrupt indicates that the IN endpoint NAK bit Set by the application has taken effect in the core. This interrupt does not guarantee that a NAK handshake is sent on the USB. A STALL bit takes priority over a NAK bit. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
5	intknepmis	IN Token Received with EP Mismatch (INTknEPMis) Applies to non-periodic IN endpoints only. Indicates that the data in the top of the non-periodic TxFIFO belongs to an endpoint other than the one For which the IN token was received. This interrupt is asserted on the endpoint For which the IN token was received. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
4	intkntxfemp	IN Token Received When TxFIFO is Empty (INTknTXFEmp) Applies to non-periodic IN endpoints only. Indicates that an IN token was received when the associated TxFIFO (periodic/non-periodic) was empty. This interrupt is asserted on the endpoint For which the IN token was received. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
3	timeout	Timeout Condition (TimeOUT) In shared TX FIFO mode, applies to non-isochronous IN endpoints only. In dedicated FIFO mode, applies only to Control IN endpoints. In Scatter/Gather DMA mode, the TimeOUT interrupt is not asserted. Indicates that the core has detected a timeout condition on the USB For the last IN token on this endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
2	ahberr	AHB Error (AHBErr) Applies to IN and OUT endpoints. This is generated only in Internal DMA mode when there is an AHB error during an AHB read/write. The application can read the corresponding endpoint DMA address register to get the error address. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
1	epdisbld	Endpoint Disabled Interrupt (EPDisbld) Applies to IN and OUT endpoints. This bit indicates that the endpoint is disabled per the application's request. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
0	xfercompl	Transfer Completed Interrupt (XferCompl) Applies to IN and OUT endpoints. When Scatter/Gather DMA mode is enabled - For IN endpoint this field indicates that the requested data from the descriptor is moved from external system memory to internal FIFO. - For OUT endpoint this field indicates that the requested data from the internal FIFO is moved to external system memory. This interrupt is generated only when the corresponding endpoint descriptor is closed, and the IOC bit For the corresponding descriptor is Set. When Scatter/Gather DMA mode is disabled, this field indicates that the programmed transfer is complete on the AHB as well as on the USB, For this endpoint. Value: Description 0x0: INACTIVE 0x1: ACTIVE	RW	0x0

dieptsiz12

Device IN Endpoint 12 Transfer Size Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00A90
i_usbotg_1_devgrp	0xFFB40800	0xFFB40A90

Offset: 0x290

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	mc RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
xfersize RW 0x0															

dieptsiz12 Fields

Bit	Name	Description	Access	Reset
30:29	mc	Applies to IN endpoints only. For periodic IN endpoints, this field indicates the number of packets that must be transmitted per microframe on the USB. The core uses this field to calculate the data PID for isochronous IN endpoints. 2'b01: 1 packet 2'b10: 2 packets 2'b11: 3 packets For non-periodic IN endpoints, this field is valid only in Internal DMA mode. It specifies the number of packets the core must fetchfor an IN endpoint before it switches to the endpoint pointed to by the Next Endpoint field of the Device Endpoint-n Control register (DIEPCTLn.NextEp) **Value** **Description** 0x1 PACKETONE 0x2 PACKETTWO 0x3 PACKETTHREE	RW	0x0

Bit	Name	Description	Access	Reset
28:19	pktcnt	Packet Count (PktCnt) Indicates the total number of USB packets that constitute the Transfer Size amount of data For endpoint 0. This field is decremented every time a packet (maximum size or short packet) is read from the TxFIFO.	RW	0x0
18:0	xfersize	Transfer Size (XferSize) Indicates the transfer size in bytes For endpoint 0. The core interrupts the application only after it has exhausted the transfer size amount of data. The transfer size can be Set to the maximum packet size of the endpoint, to be interrupted at the end of each packet. The core decrements this field every time a packet from the external memory is written to the TxFIFO.	RW	0x0

diepdma12

Device IN Endpoint 12 DMA Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00A94
i_usbotg_1_devgrp	0xFFB40800	0xFFB40A94

Offset: 0x294

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepdma12 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
diepdma12 RW 0x0															

diepdma12 Fields

Bit	Name	Description	Access	Reset
31:0	diepdma12	Holds the start address of the external memory for storing or fetching endpoint data. Note: For control endpoints, this field stores control OUT data packets as well as SETUP transaction data packets. When more than three SETUP packets are received back-to-back, the SETUP data packet in the memory is overwritten. This register is incremented on every AHB transaction. The application can give only a DWORD-aligned address. When Scatter/Gather DMA mode is not enabled, the application programs the start address value in this field. When Scatter/Gather DMA mode is enabled, this field indicates the base pointer for the descriptor list.	RW	0x0

dtxfsts12

Device IN Endpoint Transmit FIFO Status Register 12

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00A98
i_usbotg_1_devgrp	0xFFB40800	0xFFB40A98

Offset: 0x298

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ineptxfspcavail RO 0x2000															

dtxfsts12 Fields

Bit	Name	Description	Access	Reset
15:0	ineptxfspcavail	IN Endpoint TxFIFO Space Avail (INEPTxFSpcAvail) Indicates the amount of free space available in the Endpoint TxFIFO. Values are in terms of 32-bit words. 16'h0: Endpoint TxFIFO is full 16'h1: 1 word available 16'h2: 2 words available 16'hn: n words available (where 0 n 32,768) 16'h8000: 32,768 words available Others: Reserved	RO	0x2000

diepdmab12

Device IN Endpoint 12 Buffer Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00A9C
i_usbotg_1_devgrp	0xFFB40800	0xFFB40A9C

Offset: 0x29C

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepdmab12 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
diepdmab12 RO 0x0															

diepdmab12 Fields

Bit	Name	Description	Access	Reset
31:0	diepdmab12	Holds the current buffer address.This register is updated as and when the data transfer for the corresponding end point is in progress. This register is present only in Scatter/Gather DMA mode. Otherwise this field is reserved.	RO	0x0

diepctl13

Device Control IN Endpoint 13 Control Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00AA0
i_usbotg_1_devgrp	0xFFB40800	0xFFB40AA0

Offset: 0x2A0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	txfnum RW 0x0				stall RW 0x0	Reserved	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
usbactep RW 0x0	Reserved				mps RW 0x0										

diepctl13 Fields

Bit	Name	Description	Access	Reset
31	epena	Endpoint Enable (EPEna) Applies to IN and OUT endpoints. When Scatter/Gather DMA mode is enabled, For IN endpoints this bit indicates that the descriptor structure and data buffer with data ready to transmit is setup. For OUT endpoint it indicates that the descriptor structure and data buffer to receive data is setup. When Scatter/Gather DMA mode is enabledsuch as for buffer-pointer based DMA mode: - For IN endpoints, this bit indicates that data is ready to be transmitted on the endpoint. - For OUT endpoints, this bit indicates that the application has allocated the memory to start receiving data from the USB. - The core clears this bit before setting any of the following interrupts on this endpoint: SETUP Phase Done Endpoint Disabled Transfer Completed Note: For control endpoints in DMA mode, this bit must be set to be able to transfer SETUP data packets in memory. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
30	epdis	Endpoint Disable (EPDis) Applies to IN and OUT endpoints. The application sets this bit to stop transmitting/receiving data on an endpoint, even before the transfer for that endpoint is complete. The application must wait for the Endpoint Disabled interrupt before treating the endpoint as disabled. The core clears this bit before setting the Endpoint Disabled interrupt. The application must set this bit only if Endpoint Enable is already set for this endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
29	setd1pid	Set DATA1 PID (SetD1PID) Applies to interrupt/bulk IN and OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA1. This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. Set Odd (micro)frame (SetOddFr) Applies to isochronous IN and OUT endpoints only. Writing to this field sets the Even/Odd (micro)frame (EO_FrNum) field to odd (micro)frame. This field is not applicable for Scatter/Gather DMA mode. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	WO	0x0

Bit	Name	Description	Access	Reset
28	setd0pid	Set DATA0 PID (SetD0PID) Applies to interrupt/bulk IN and OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA0. This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. 1'b0 WO In non-Scatter/Gather DMA mode: Set Even (micro)frame (SetEvenFr) Applies to isochronous IN and OUT endpoints only. Writing to this field sets the Even/Odd (micro)frame (EO_FrNum) field to even (micro) frame. When Scatter/Gather DMA mode is enabled, this field is reserved. The frame number in which to send data is in the transmit descriptor structure. The frame in which to receive data is updated in receive descriptor structure. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	WO	0x0
27	snak	Set NAK (SNAK) A write to this bit sets the NAK bit For the endpoint. Using this bit, the application can control the transmission of NAK handshakes on an endpoint. The core can also Set this bit For an endpoint after a SETUP packet is received on that endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	WO	0x0

Bit	Name	Description	Access	Reset
26	cnak	Clear NAK (CNAK) A write to this bit clears the NAK bit For the endpoint. Value: Description 0x0: INACTIVE 0x1: ACTIVE	WO	0x0
25:22	txfnum	TxFIFO Number (TxFNum) Shared FIFO Operation non-periodic endpoints must set this bit to zero. Periodic endpoints must map this to the corresponding Periodic TxFIFO number. 4'h0: Non-Periodic TxFIFO Others: Specified Periodic TxFIFO.number Note: An interrupt IN endpoint can be configured as a non-periodic endpoint for applications such as mass storage. The core treats an IN endpoint as a non-periodic endpoint if the TxFNum field is set to 0. Otherwise, a separate periodic FIFO must be allocated for an interrupt IN endpoint, and the number of this FIFO must be programmed into the TxFNum field. Configuring an interrupt IN endpoint as a non-periodic endpoint saves the extra periodic FIFO area. Dedicated FIFO Operationthese bits specify the FIFO number associated with this endpoint. Each active IN endpoint must be programmed to a separate FIFO number. This field is valid only for IN endpoints.	RW	0x0

Bit	Name	Description	Access	Reset
21	stall	STALL Handshake (Stall) Applies to non-control, non-isochronous IN and OUT endpoints only. The application sets this bit to stall all tokens from the USB host to this endpoint. If a NAK bit, Global Non-periodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority. Only the application can clear this bit, never the core. 1'b0 R_W Applies to control endpoints only. The application can only set this bit, and the core clears it, when a SETUP token is received for this endpoint. If a NAK bit, Global Non-periodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority. Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake. **Value: Description** 0x0: INACTIVE 0x1: ACTIVE	RW	0x0
19:18	eptype	Endpoint Type (EPType) This is the transfer type supported by this logical endpoint. 2'b00: Control 2'b01: Isochronous 2'b10: Bulk 2'b11: Interrupt **Value: Description** 0x0: CONTROL 0x1: ISOCHRONOUS 0x2: BULK 0x3: INTERRUP	RW	0x0

Bit	Name	Description	Access	Reset
17	naksts	NAK Status (NAKSts) Indicates the following: 1'b0: The core is transmitting non-NAK handshakes based on the FIFO status. 1'b1: The core is transmitting NAK handshakes on this endpoint. When either the application or the core sets this bit: The core stops receiving any data on an OUT endpoint, even if there is space in the RxFIFO to accommodate the incoming packet. For non-isochronous IN endpoints: The core stops transmitting any data on an IN endpoint, even if there data is available in the TxFIFO. For isochronous IN endpoints: The core sends out a zero-length data packet, even if there data is available in the TxFIFO. Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake. **Value** **Description** 0x0 NONNAK 0x1 NAK	RO	0x0

Bit	Name	Description	Access	Reset
16	dpid	Endpoint Data PID (DPID) Applies to interrupt/bulk IN and OUT endpoints only. Contains the PID of the packet to be received or transmitted on this endpoint. The application must program the PID of the first packet to be received or transmitted on this endpoint, after the endpoint is activated. The applications use the SetD1PID and SetD0PID fields of this register to program either DATA0 or DATA1 PID. 1'b0: DATA0 1'b1: DATA1 This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. 1'b0 RO Even/Odd (Micro)Frame (EO_FrNum) In non-Scatter/Gather DMA mode: Applies to isochronous IN and OUT endpoints only. Indicates the (micro)frame number in which the core transmits/ receives isochronous data for this endpoint. The application must program the even/odd (micro) frame number in which it intends to transmit/receive isochronous data for this endpoint using the SetEvnFr and SetOddFr fields in this register. 1'b0: Even (micro)frame 1'b1: Odd (micro)frame When Scatter/Gather DMA mode is enabled, this field is reserved. The frame number in which to send data is provided in the transmit descriptor structure. The frame in which data is received is updated in receive descriptor structure. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RO	0x0

Bit	Name	Description	Access	Reset
15	usbactep	USB Active Endpoint (USBActEP) Indicates whether this endpoint is active in the current configuration and interface. The core clears this bit for all endpoints (other than EP 0) after detecting a USB reset. After receiving the SetConfiguration and SetInterface commands, the applica-tion must program endpoint registers accordingly and set this bit. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
10:0	mps	Maximum Packet Size (MPS) The application must program this field with the maximum packet size for the current logical endpoint. This value is in bytes.	RW	0x0

diepint13

Device IN Endpoint 13 Interrupt Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00AA8
i_usbotg_1_devgrp	0xFFB40800	0xFFB40AA8

Offset: 0x2A8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	txfifoundrn RW 0x0	txfempp RO 0x1	inepnakeff RW 0x0	intknepmis RW 0x0	intkntxfempp RW 0x0	timeout RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0

diepint13 Fields

Bit	Name	Description	Access	Reset
14	nyetintrpt	NYET Interrupt (NYETIntrpt) The core generates this interrupt when a NYET response is transmitted for a non isochronous OUT endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
13	nakintrpt	NAK Interrupt (NAKInterrupt) The core generates this interrupt when a NAK is transmitted or received by the device. In case of isochronous IN endpoints the interrupt gets generated when a zero length packet is transmitted due to un-availability of data in the TXFifo. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
12	bbleerr	NAK Interrupt (BbleErr) The core generates this interrupt when babble is received for the endpoint. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
11	pktdrpsts	Packet Drop Status (PktDrpSts) This bit indicates to the application that an ISOC OUT packet has been dropped. This bit does not have an associated mask bit and does not generate an interrupt. Dependency: This bit is valid in non Scatter/Gather DMA mode when periodic transfer interrupt feature is selected. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
9	bnaintr	BNA (Buffer Not Available) Interrupt (BNAIntr) This bit is valid only when Scatter/Gather DMA mode is enabled. The core generates this interrupt when the descriptor accessed is not ready For the Core to process, such as Host busy or DMA done **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
8	txfifoundrn	Fifo Underrun (TxfifoUndrn) Applies to IN endpoints Only This bit is valid only If thresholding is enabled. The core generates this interrupt when it detects a transmit FIFO underrun condition For this endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
7	txfemp	Transmit FIFO Empty (TxFEmp) This bit is valid only For IN Endpoints This interrupt is asserted when the TxFIFO For this endpoint is either half or completely empty. The half or completely empty status is determined by the TxFIFO Empty Level bit in the Core AHB Configuration register (GAHBCFG.NPTxFEmpLvl)). **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RO	0x1

Bit	Name	Description	Access	Reset
6	inepnakeff	IN Endpoint NAK Effective (INEPNakEff) Applies to periodic IN endpoints only. This bit can be cleared when the application clears the IN endpoint NAK by writing to DIEPCTLn.CNAK. This interrupt indicates that the core has sampled the NAK bit Set (either by the application or by the core). The interrupt indicates that the IN endpoint NAK bit Set by the application has taken effect in the core. This interrupt does not guarantee that a NAK handshake is sent on the USB. A STALL bit takes priority over a NAK bit. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
5	intknepmis	IN Token Received with EP Mismatch (INTknEPMis) Applies to non-periodic IN endpoints only. Indicates that the data in the top of the non-periodic TxFIFO belongs to an endpoint other than the one For which the IN token was received. This interrupt is asserted on the endpoint For which the IN token was received. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
4	intkntxfemp	IN Token Received When TxFIFO is Empty (INTknTXFEmp) Applies to non-periodic IN endpoints only. Indicates that an IN token was received when the associated TxFIFO (periodic/non-periodic) was empty. This interrupt is asserted on the endpoint For which the IN token was received. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
3	timeout	Timeout Condition (TimeOUT) In shared TX FIFO mode, applies to non-isochronous IN endpoints only. In dedicated FIFO mode, applies only to Control IN endpoints. In Scatter/Gather DMA mode, the TimeOUT interrupt is not asserted. Indicates that the core has detected a timeout condition on the USB For the last IN token on this endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
2	ahberr	AHB Error (AHBErr) Applies to IN and OUT endpoints. This is generated only in Internal DMA mode when there is an AHB error during an AHB read/write. The application can read the corresponding endpoint DMA address register to get the error address. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
1	epdisbld	Endpoint Disabled Interrupt (EPDisbld) Applies to IN and OUT endpoints. This bit indicates that the endpoint is disabled per the application's request. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
0	xfercompl	Transfer Completed Interrupt (XferCompl) Applies to IN and OUT endpoints. When Scatter/Gather DMA mode is enabled - For IN endpoint this field indicates that the requested data from the descriptor is moved from external system memory to internal FIFO. - For OUT endpoint this field indicates that the requested data from the internal FIFO is moved to external system memory. This interrupt is generated only when the corresponding endpoint descriptor is closed, and the IOC bit For the corresponding descriptor is Set. When Scatter/Gather DMA mode is disabled, this field indicates that the programmed transfer is complete on the AHB as well as on the USB, For this endpoint. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

dieptsiz13

Device IN Endpoint 13 Transfer Size Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00AB0
i_usbotg_1_devgrp	0xFFB40800	0xFFB40AB0

Offset: 0x2B0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	mc RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
xfersize RW 0x0															

dieptsiz13 Fields

Bit	Name	Description	Access	Reset
30:29	mc	Applies to IN endpoints only. For periodic IN endpoints, this field indicates the number of packets that must be transmitted per microframe on the USB. The core uses this field to calculate the data PID for isochronous IN endpoints. 2'b01: 1 packet 2'b10: 2 packets 2'b11: 3 packets For non-periodic IN endpoints, this field is valid only in Internal DMA mode. It specifies the number of packets the core must fetchfor an IN endpoint before it switches to the endpoint pointed to by the Next Endpoint field of the Device Endpoint-n Control register (DIEPCTLn.NextEp) **Value** — **Description** 0x1 — PACKETONE 0x2 — PACKETTWO 0x3 — PACKETTHREE	RW	0x0

Bit	Name	Description	Access	Reset
28:19	pktcnt	Packet Count (PktCnt) Indicates the total number of USB packets that constitute the Transfer Size amount of data For endpoint 0. This field is decremented every time a packet (maximum size or short packet) is read from the TxFIFO.	RW	0x0
18:0	xfersize	Transfer Size (XferSize) Indicates the transfer size in bytes For endpoint 0. The core interrupts the application only after it has exhausted the transfer size amount of data. The transfer size can be Set to the maximum packet size of the endpoint, to be interrupted at the end of each packet. The core decrements this field every time a packet from the external memory is written to the TxFIFO.	RW	0x0

diepdma13

Device IN Endpoint 13 DMA Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00AB4
i_usbotg_1_devgrp	0xFFB40800	0xFFB40AB4

Offset: 0x2B4

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepdma13 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
diepdma13 RW 0x0															

diepdma13 Fields

Bit	Name	Description	Access	Reset
31:0	diepdma13	Holds the start address of the external memory for storing or fetching endpoint data. Note: For control endpoints, this field stores control OUT data packets as well as SETUP transaction data packets. When more than three SETUP packets are received back-to-back, the SETUP data packet in the memory is overwritten. This register is incremented on every AHB transaction. The application can give only a DWORD-aligned address. When Scatter/Gather DMA mode is not enabled, the application programs the start address value in this field. When Scatter/Gather DMA mode is enabled, this field indicates the base pointer for the descriptor list.	RW	0x0

dtxfsts13

Device IN Endpoint Transmit FIFO Status Register 13

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00AB8
i_usbotg_1_devgrp	0xFFB40800	0xFFB40AB8

Offset: 0x2B8

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ineptxfspcavail RO 0x2000															

dtxfsts13 Fields

Bit	Name	Description	Access	Reset
15:0	ineptxfspcavail	IN Endpoint TxFIFO Space Avail (INEPTxFSpcAvail) Indicates the amount of free space available in the Endpoint TxFIFO. Values are in terms of 32-bit words. 16'h0: Endpoint TxFIFO is full 16'h1: 1 word available 16'h2: 2 words available 16'hn: n words available (where 0 n 32,768) 16'h8000: 32,768 words available Others: Reserved	RO	0x2000

diepdmab13

Device IN Endpoint 13 Buffer Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00ABC
i_usbotg_1_devgrp	0xFFB40800	0xFFB40ABC

Offset: 0x2BC

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepdmab13 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
diepdmab13 RO 0x0															

diepdmab13 Fields

Bit	Name	Description	Access	Reset
31:0	diepdmab13	Holds the current buffer address.This register is updated as and when the data transfer for the corresponding end point is in progress. This register is present only in Scatter/Gather DMA mode. Otherwise this field is reserved.	RO	0x0

diepctl14

Device Control IN Endpoint 14 Control Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00AC0
i_usbotg_1_devgrp	0xFFB40800	0xFFB40AC0

Offset: 0x2C0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	txfnum RW 0x0				stall RW 0x0	Reserved	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
usbactep RW 0x0	Reserved				mps RW 0x0										

diepctl14 Fields

Bit	Name	Description	Access	Reset
31	epena	Endpoint Enable (EPEna) Applies to IN and OUT endpoints. When Scatter/Gather DMA mode is enabled, For IN endpoints this bit indicates that the descriptor structure and data buffer with data ready to transmit is setup. For OUT endpoint it indicates that the descriptor structure and data buffer to receive data is setup. When Scatter/Gather DMA mode is enabledsuch as for buffer-pointer based DMA mode: - For IN endpoints, this bit indicates that data is ready to be transmitted on the endpoint. - For OUT endpoints, this bit indicates that the application has allocated the memory to start receiving data from the USB. - The core clears this bit before setting any of the following interrupts on this endpoint: SETUP Phase Done Endpoint Disabled Transfer Completed Note: For control endpoints in DMA mode, this bit must be set to be able to transfer SETUP data packets in memory. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
30	epdis	Endpoint Disable (EPDis) Applies to IN and OUT endpoints. The application sets this bit to stop transmitting/receiving data on an endpoint, even before the transfer for that endpoint is complete. The application must wait for the Endpoint Disabled interrupt before treating the endpoint as disabled. The core clears this bit before setting the Endpoint Disabled interrupt. The application must set this bit only if Endpoint Enable is already set for this endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
29	setd1pid	Set DATA1 PID (SetD1PID) Applies to interrupt/bulk IN and OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA1. This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. Set Odd (micro)frame (SetOddFr) Applies to isochronous IN and OUT endpoints only. Writing to this field sets the Even/Odd (micro)frame (EO_FrNum) field to odd (micro)frame. This field is not applicable for Scatter/Gather DMA mode. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	WO	0x0

Bit	Name	Description	Access	Reset
28	setd0pid	Set DATA0 PID (SetD0PID) Applies to interrupt/bulk IN and OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA0. This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. 1'b0 WO In non-Scatter/Gather DMA mode: Set Even (micro)frame (SetEvenFr) Applies to isochronous IN and OUT endpoints only. Writing to this field sets the Even/Odd (micro)frame (EO_FrNum) field to even (micro) frame. When Scatter/Gather DMA mode is enabled, this field is reserved. The frame number in which to send data is in the transmit descriptor structure. The frame in which to receive data is updated in receive descriptor structure. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	WO	0x0
27	snak	Set NAK (SNAK) A write to this bit sets the NAK bit For the endpoint. Using this bit, the application can control the transmission of NAK handshakes on an endpoint. The core can also Set this bit For an endpoint after a SETUP packet is received on that endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	WO	0x0

Bit	Name	Description	Access	Reset
26	cnak	Clear NAK (CNAK) A write to this bit clears the NAK bit For the endpoint. Value: 0x0 — Description: INACTIVE Value: 0x1 — Description: ACTIVE	WO	0x0
25:22	txfnum	TxFIFO Number (TxFNum) Shared FIFO Operation non-periodic endpoints must set this bit to zero. Periodic endpoints must map this to the corresponding Periodic TxFIFO number. 4'h0: Non-Periodic TxFIFO Others: Specified Periodic TxFIFO.number Note: An interrupt IN endpoint can be configured as a non-periodic endpoint for applications such as mass storage. The core treats an IN endpoint as a non-periodic endpoint if the TxFNum field is set to 0. Otherwise, a separate periodic FIFO must be allocated for an interrupt IN endpoint, and the number of this FIFO must be programmed into the TxFNum field. Configuring an interrupt IN endpoint as a non-periodic endpoint saves the extra periodic FIFO area. Dedicated FIFO Operationthese bits specify the FIFO number associated with this endpoint. Each active IN endpoint must be programmed to a separate FIFO number. This field is valid only for IN endpoints.	RW	0x0

Bit	Name	Description	Access	Reset
21	stall	STALL Handshake (Stall) Applies to non-control, non-isochronous IN and OUT endpoints only. The application sets this bit to stall all tokens from the USB host to this endpoint. If a NAK bit, Global Non-periodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority. Only the application can clear this bit, never the core. 1'b0 R_W Applies to control endpoints only. The application can only set this bit, and the core clears it, when a SETUP token is received for this endpoint. If a NAK bit, Global Non-periodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority. Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake. **Value: Description** 0x0: INACTIVE 0x1: ACTIVE	RW	0x0
19:18	eptype	Endpoint Type (EPType) This is the transfer type supported by this logical endpoint. 2'b00: Control 2'b01: Isochronous 2'b10: Bulk 2'b11: Interrupt **Value: Description** 0x0: CONTROL 0x1: ISOCHRONOUS 0x2: BULK 0x3: INTERRUP	RW	0x0

Bit	Name	Description	Access	Reset
17	naksts	NAK Status (NAKSts) Indicates the following: 1'b0: The core is transmitting non-NAK handshakes based on the FIFO status. 1'b1: The core is transmitting NAK handshakes on this endpoint. When either the application or the core sets this bit: The core stops receiving any data on an OUT endpoint, even if there is space in the RxFIFO to accommodate the incoming packet. For non-isochronous IN endpoints: The core stops transmitting any data on an IN endpoint, even if there data is available in the TxFIFO. For isochronous IN endpoints: The core sends out a zero-length data packet, even if there data is available in the TxFIFO. Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake. **Value** — **Description** 0x0 — NONNAK 0x1 — NAK	RO	0x0

Bit	Name	Description	Access	Reset
16	dpid	Endpoint Data PID (DPID) Applies to interrupt/bulk IN and OUT endpoints only. Contains the PID of the packet to be received or transmitted on this endpoint. The application must program the PID of the first packet to be received or transmitted on this endpoint, after the endpoint is activated. The applications use the SetD1PID and SetD0PID fields of this register to program either DATA0 or DATA1 PID. 1'b0: DATA0 1'b1: DATA1 This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. 1'b0 RO Even/Odd (Micro)Frame (EO_FrNum) In non-Scatter/Gather DMA mode: Applies to isochronous IN and OUT endpoints only. Indicates the (micro)frame number in which the core transmits/receives isochronous data for this endpoint. The application must program the even/odd (micro) frame number in which it intends to transmit/receive isochronous data for this endpoint using the SetEvnFr and SetOddFr fields in this register. 1'b0: Even (micro)frame 1'b1: Odd (micro)frame When Scatter/Gather DMA mode is enabled, this field is reserved. The frame number in which to send data is provided in the transmit descriptor structure. The frame in which data is received is updated in receive descriptor structure. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RO	0x0

Bit	Name	Description	Access	Reset
15	usbactep	USB Active Endpoint (USBActEP) Indicates whether this endpoint is active in the current configuration and interface. The core clears this bit for all endpoints (other than EP 0) after detecting a USB reset. After receiving the SetConfiguration and SetInterface commands, the applica-tion must program endpoint registers accordingly and set this bit. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
10:0	mps	Maximum Packet Size (MPS) The application must program this field with the maximum packet size for the current logical endpoint. This value is in bytes.	RW	0x0

diepint14

Device IN Endpoint 14 Interrupt Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00AC8
i_usbotg_1_devgrp	0xFFB40800	0xFFB40AC8

Offset: 0x2C8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	txfifoundrn RW 0x0	txfempp RO 0x1	inepnakeff RW 0x0	intknepmis RW 0x0	intkntxfempp RW 0x0	timeout RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0

diepint14 Fields

Bit	Name	Description	Access	Reset
14	nyetintrpt	NYET Interrupt (NYETIntrpt) The core generates this interrupt when a NYET response is transmitted for a non isochronous OUT endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
13	nakintrpt	NAK Interrupt (NAKInterrupt) The core generates this interrupt when a NAK is transmitted or received by the device. In case of isochronous IN endpoints the interrupt gets generated when a zero length packet is transmitted due to un-availability of data in the TXFifo. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
12	bbleerr	NAK Interrupt (BbleErr) The core generates this interrupt when babble is received for the endpoint. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
11	pktdrpsts	Packet Drop Status (PktDrpSts) This bit indicates to the application that an ISOC OUT packet has been dropped. This bit does not have an associated mask bit and does not generate an interrupt. Dependency: This bit is valid in non Scatter/Gather DMA mode when periodic transfer interrupt feature is selected. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
9	bnaintr	BNA (Buffer Not Available) Interrupt (BNAIntr) This bit is valid only when Scatter/Gather DMA mode is enabled. The core generates this interrupt when the descriptor accessed is not ready For the Core to process, such as Host busy or DMA done **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
8	txfifoundrn	Fifo Underrun (TxfifoUndrn) Applies to IN endpoints Only This bit is valid only If thresholding is enabled. The core generates this interrupt when it detects a transmit FIFO underrun condition For this endpoint. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
7	txfemp	Transmit FIFO Empty (TxFEmp) This bit is valid only For IN Endpoints This interrupt is asserted when the TxFIFO For this endpoint is either half or completely empty. The half or completely empty status is determined by the TxFIFO Empty Level bit in the Core AHB Configuration register (GAHBCFG.NPTxFEmpLvl)). **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x1

Bit	Name	Description	Access	Reset
6	inepnakeff	IN Endpoint NAK Effective (INEPNakEff) Applies to periodic IN endpoints only. This bit can be cleared when the application clears the IN endpoint NAK by writing to DIEPCTLn.CNAK. This interrupt indicates that the core has sampled the NAK bit Set (either by the application or by the core). The interrupt indicates that the IN endpoint NAK bit Set by the application has taken effect in the core. This interrupt does not guarantee that a NAK handshake is sent on the USB. A STALL bit takes priority over a NAK bit. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
5	intknepmis	IN Token Received with EP Mismatch (INTknEPMis) Applies to non-periodic IN endpoints only. Indicates that the data in the top of the non-periodic TxFIFO belongs to an endpoint other than the one For which the IN token was received. This interrupt is asserted on the endpoint For which the IN token was received. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
4	intkntxfemp	IN Token Received When TxFIFO is Empty (INTknTXFEmp) Applies to non-periodic IN endpoints only. Indicates that an IN token was received when the associated TxFIFO (periodic/non-periodic) was empty. This interrupt is asserted on the endpoint For which the IN token was received. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
3	timeout	Timeout Condition (TimeOUT) In shared TX FIFO mode, applies to non-isochronous IN endpoints only. In dedicated FIFO mode, applies only to Control IN endpoints. In Scatter/Gather DMA mode, the TimeOUT interrupt is not asserted. Indicates that the core has detected a timeout condition on the USB For the last IN token on this endpoint. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
2	ahberr	AHB Error (AHBErr) Applies to IN and OUT endpoints. This is generated only in Internal DMA mode when there is an AHB error during an AHB read/write. The application can read the corresponding endpoint DMA address register to get the error address. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
1	epdisbld	Endpoint Disabled Interrupt (EPDisbld) Applies to IN and OUT endpoints. This bit indicates that the endpoint is disabled per the application's request. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
0	xfercompl	Transfer Completed Interrupt (XferCompl) Applies to IN and OUT endpoints. When Scatter/Gather DMA mode is enabled - For IN endpoint this field indicates that the requested data from the descriptor is moved from external system memory to internal FIFO. - For OUT endpoint this field indicates that the requested data from the internal FIFO is moved to external system memory. This interrupt is generated only when the corresponding endpoint descriptor is closed, and the IOC bit For the corresponding descriptor is Set. When Scatter/Gather DMA mode is disabled, this field indicates that the programmed transfer is complete on the AHB as well as on the USB, For this endpoint. Value: 0x0 — INACTIVE; 0x1 — ACTIVE	RW	0x0

dieptsiz14

Device IN Endpoint 14 Transfer Size Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00AD0
i_usbotg_1_devgrp	0xFFB40800	0xFFB40AD0

Offset: 0x2D0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	mc RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
xfersize RW 0x0															

dieptsiz14 Fields

Bit	Name	Description	Access	Reset
30:29	mc	Applies to IN endpoints only. For periodic IN endpoints, this field indicates the number of packets that must be transmitted per microframe on the USB. The core uses this field to calculate the data PID for isochronous IN endpoints. 2'b01: 1 packet 2'b10: 2 packets 2'b11: 3 packets For non-periodic IN endpoints, this field is valid only in Internal DMA mode. It specifies the number of packets the core must fetchfor an IN endpoint before it switches to the endpoint pointed to by the Next Endpoint field of the Device Endpoint-n Control register (DIEPCTLn.NextEp) **Value** — **Description** 0x1 — PACKETONE 0x2 — PACKETTWO 0x3 — PACKETTHREE	RW	0x0

Bit	Name	Description	Access	Reset
28:19	pktcnt	Packet Count (PktCnt) Indicates the total number of USB packets that constitute the Transfer Size amount of data For endpoint 0. This field is decremented every time a packet (maximum size or short packet) is read from the TxFIFO.	RW	0x0
18:0	xfersize	Transfer Size (XferSize) Indicates the transfer size in bytes For endpoint 0. The core interrupts the application only after it has exhausted the transfer size amount of data. The transfer size can be Set to the maximum packet size of the endpoint, to be interrupted at the end of each packet. The core decrements this field every time a packet from the external memory is written to the TxFIFO.	RW	0x0

diepdma14

Device IN Endpoint 14 DMA Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00AD4
i_usbotg_1_devgrp	0xFFB40800	0xFFB40AD4

Offset: 0x2D4

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepdma14 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
diepdma14 RW 0x0															

diepdma14 Fields

Bit	Name	Description	Access	Reset
31:0	diepdma14	Holds the start address of the external memory for storing or fetching endpoint data. Note: For control endpoints, this field stores control OUT data packets as well as SETUP transaction data packets. When more than three SETUP packets are received back-to-back, the SETUP data packet in the memory is overwritten. This register is incremented on every AHB transaction. The application can give only a DWORD-aligned address. When Scatter/Gather DMA mode is not enabled, the application programs the start address value in this field. When Scatter/Gather DMA mode is enabled, this field indicates the base pointer for the descriptor list.	RW	0x0

dtxfsts14

Device IN Endpoint Transmit FIFO Status Register 14

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00AD8
i_usbotg_1_devgrp	0xFFB40800	0xFFB40AD8

Offset: 0x2D8

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ineptxfspcavail RO 0x2000															

dtxfsts14 Fields

Bit	Name	Description	Access	Reset
15:0	ineptxfspcavail	IN Endpoint TxFIFO Space Avail (INEPTxFSpcAvail) Indicates the amount of free space available in the Endpoint TxFIFO. Values are in terms of 32-bit words. 16'h0: Endpoint TxFIFO is full 16'h1: 1 word available 16'h2: 2 words available 16'hn: n words available (where 0 n 32,768) 16'h8000: 32,768 words available Others: Reserved	RO	0x2000

diepdmab14

Device IN Endpoint 14 Buffer Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00ADC
i_usbotg_1_devgrp	0xFFB40800	0xFFB40ADC

Offset: 0x2DC

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepdmab14 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
diepdmab14 RO 0x0															

diepdmab14 Fields

Bit	Name	Description	Access	Reset
31:0	diepdmab14	Holds the current buffer address.This register is updated as and when the data transfer for the corresponding end point is in progress. This register is present only in Scatter/Gather DMA mode. Otherwise this field is reserved.	RO	0x0

diepctl15

Device Control IN Endpoint 15 Control Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00AE0
i_usbotg_1_devgrp	0xFFB40800	0xFFB40AE0

Offset: 0x2E0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	txfnum RW 0x0				stall RW 0x0	Reserved	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
usbactep RW 0x0	Reserved				mps RW 0x0										

diepctl15 Fields

Bit	Name	Description	Access	Reset
31	epena	Endpoint Enable (EPEna) Applies to IN and OUT endpoints. When Scatter/Gather DMA mode is enabled, For IN endpoints this bit indicates that the descriptor structure and data buffer with data ready to transmit is setup. For OUT endpoint it indicates that the descriptor structure and data buffer to receive data is setup. When Scatter/Gather DMA mode is enabledsuch as for buffer-pointer based DMA mode: - For IN endpoints, this bit indicates that data is ready to be transmitted on the endpoint. - For OUT endpoints, this bit indicates that the application has allocated the memory to start receiving data from the USB. - The core clears this bit before setting any of the following interrupts on this endpoint: SETUP Phase Done Endpoint Disabled Transfer Completed Note: For control endpoints in DMA mode, this bit must be set to be able to transfer SETUP data packets in memory. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
30	epdis	Endpoint Disable (EPDis) Applies to IN and OUT endpoints. The application sets this bit to stop transmitting/receiving data on an endpoint, even before the transfer for that endpoint is complete. The application must wait for the Endpoint Disabled interrupt before treating the endpoint as disabled. The core clears this bit before setting the Endpoint Disabled interrupt. The application must set this bit only if Endpoint Enable is already set for this endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
29	setd1pid	Set DATA1 PID (SetD1PID) Applies to interrupt/bulk IN and OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA1. This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. Set Odd (micro)frame (SetOddFr) Applies to isochronous IN and OUT endpoints only. Writing to this field sets the Even/Odd (micro)frame (EO_FrNum) field to odd (micro)frame. This field is not applicable for Scatter/Gather DMA mode. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	WO	0x0

Bit	Name	Description	Access	Reset
28	setd0pid	Set DATA0 PID (SetD0PID) Applies to interrupt/bulk IN and OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA0. This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. 1'b0 WO In non-Scatter/Gather DMA mode: Set Even (micro)frame (SetEvenFr) Applies to isochronous IN and OUT endpoints only. Writing to this field sets the Even/Odd (micro)frame (EO_FrNum) field to even (micro) frame. When Scatter/Gather DMA mode is enabled, this field is reserved. The frame number in which to send data is in the transmit descriptor structure. The frame in which to receive data is updated in receive descriptor structure. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	WO	0x0
27	snak	Set NAK (SNAK) A write to this bit sets the NAK bit For the endpoint. Using this bit, the application can control the transmission of NAK handshakes on an endpoint. The core can also Set this bit For an endpoint after a SETUP packet is received on that endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	WO	0x0

Bit	Name	Description	Access	Reset
26	cnak	Clear NAK (CNAK) A write to this bit clears the NAK bit For the endpoint. Value: 0x0 — INACTIVE; 0x1 — ACTIVE	WO	0x0
25:22	txfnum	TxFIFO Number (TxFNum) Shared FIFO Operation non-periodic endpoints must set this bit to zero. Periodic endpoints must map this to the corresponding Periodic TxFIFO number. 4'h0: Non-Periodic TxFIFO Others: Specified Periodic TxFIFO.number Note: An interrupt IN endpoint can be configured as a non-periodic endpoint for applications such as mass storage. The core treats an IN endpoint as a non-periodic endpoint if the TxFNum field is set to 0. Otherwise, a separate periodic FIFO must be allocated for an interrupt IN endpoint, and the number of this FIFO must be programmed into the TxFNum field. Configuring an interrupt IN endpoint as a non-periodic endpoint saves the extra periodic FIFO area. Dedicated FIFO Operationthese bits specify the FIFO number associated with this endpoint. Each active IN endpoint must be programmed to a separate FIFO number. This field is valid only for IN endpoints.	RW	0x0

Bit	Name	Description	Access	Reset
21	stall	STALL Handshake (Stall) Applies to non-control, non-isochronous IN and OUT endpoints only. The application sets this bit to stall all tokens from the USB host to this endpoint. If a NAK bit, Global Non-periodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority. Only the application can clear this bit, never the core. 1'b0 R_W Applies to control endpoints only. The application can only set this bit, and the core clears it, when a SETUP token is received for this endpoint. If a NAK bit, Global Non-periodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority. Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake. **Value: Description** 0x0: INACTIVE 0x1: ACTIVE	RW	0x0
19:18	eptype	Endpoint Type (EPType) This is the transfer type supported by this logical endpoint. 2'b00: Control 2'b01: Isochronous 2'b10: Bulk 2'b11: Interrupt **Value: Description** 0x0: CONTROL 0x1: ISOCHRONOUS 0x2: BULK 0x3: INTERRUP	RW	0x0

Bit	Name	Description	Access	Reset
17	naksts	NAK Status (NAKSts) Indicates the following: 1'b0: The core is transmitting non-NAK handshakes based on the FIFO status. 1'b1: The core is transmitting NAK handshakes on this endpoint. When either the application or the core sets this bit: The core stops receiving any data on an OUT endpoint, even if there is space in the RxFIFO to accommodate the incoming packet. For non-isochronous IN endpoints: The core stops transmitting any data on an IN endpoint, even if there data is available in the TxFIFO. For isochronous IN endpoints: The core sends out a zero-length data packet, even if there data is available in the TxFIFO. Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake. **Value** — **Description** 0x0 — NONNAK 0x1 — NAK	RO	0x0

Bit	Name	Description	Access	Reset
16	dpid	Endpoint Data PID (DPID) Applies to interrupt/bulk IN and OUT endpoints only. Contains the PID of the packet to be received or transmitted on this endpoint. The application must program the PID of the first packet to be received or transmitted on this endpoint, after the endpoint is activated. The applications use the SetD1PID and SetD0PID fields of this register to program either DATA0 or DATA1 PID. 1'b0: DATA0 1'b1: DATA1 This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. 1'b0 RO Even/Odd (Micro)Frame (EO_FrNum) In non-Scatter/Gather DMA mode: Applies to isochronous IN and OUT endpoints only. Indicates the (micro)frame number in which the core transmits/ receives isochronous data for this endpoint. The application must program the even/odd (micro) frame number in which it intends to transmit/receive isochronous data for this endpoint using the SetEvnFr and SetOddFr fields in this register. 1'b0: Even (micro)frame 1'b1: Odd (micro)frame When Scatter/Gather DMA mode is enabled, this field is reserved. The frame number in which to send data is provided in the transmit descriptor structure. The frame in which data is received is updated in receive descriptor structure. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RO	0x0

Bit	Name	Description	Access	Reset
15	usbactep	USB Active Endpoint (USBActEP) Indicates whether this endpoint is active in the current configuration and interface. The core clears this bit for all endpoints (other than EP 0) after detecting a USB reset. After receiving the SetConfiguration and SetInterface commands, the applica-tion must program endpoint registers accordingly and set this bit. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
10:0	mps	Maximum Packet Size (MPS) The application must program this field with the maximum packet size for the current logical endpoint. This value is in bytes.	RW	0x0

diepint15

Device IN Endpoint 15 Interrupt Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00AE8
i_usbotg_1_devgrp	0xFFB40800	0xFFB40AE8

Offset: 0x2E8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	txfifoundrn RW 0x0	txfemp RO 0x1	inepnakeff RW 0x0	intknepmis RW 0x0	intkntxfemp RW 0x0	timeout RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0

diepint15 Fields

Bit	Name	Description	Access	Reset
14	nyetintrpt	NYET Interrupt (NYETIntrpt) The core generates this interrupt when a NYET response is transmitted for a non isochronous OUT endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
13	nakintrpt	NAK Interrupt (NAKInterrupt) The core generates this interrupt when a NAK is transmitted or received by the device. In case of isochronous IN endpoints the interrupt gets generated when a zero length packet is transmitted due to un-availability of data in the TXFifo. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
12	bbleerr	NAK Interrupt (BbleErr) The core generates this interrupt when babble is received for the endpoint. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
11	pktdrpsts	Packet Drop Status (PktDrpSts) This bit indicates to the application that an ISOC OUT packet has been dropped. This bit does not have an associated mask bit and does not generate an interrupt. Dependency: This bit is valid in non Scatter/Gather DMA mode when periodic transfer interrupt feature is selected. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
9	bnaintr	BNA (Buffer Not Available) Interrupt (BNAIntr) This bit is valid only when Scatter/Gather DMA mode is enabled. The core generates this interrupt when the descriptor accessed is not ready For the Core to process, such as Host busy or DMA done **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
8	txfifoundrn	Fifo Underrun (TxfifoUndrn) Applies to IN endpoints Only This bit is valid only If thresholding is enabled. The core generates this interrupt when it detects a transmit FIFO underrun condition For this endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
7	txfemp	Transmit FIFO Empty (TxFEmp) This bit is valid only For IN Endpoints This interrupt is asserted when the TxFIFO For this endpoint is either half or completely empty. The half or completely empty status is determined by the TxFIFO Empty Level bit in the Core AHB Configuration register (GAHBCFG.NPTxFEmpLvl)). **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RO	0x1

Bit	Name	Description	Access	Reset
6	inepnakeff	IN Endpoint NAK Effective (INEPNakEff) Applies to periodic IN endpoints only. This bit can be cleared when the application clears the IN endpoint NAK by writing to DIEPCTLn.CNAK. This interrupt indicates that the core has sampled the NAK bit Set (either by the application or by the core). The interrupt indicates that the IN endpoint NAK bit Set by the application has taken effect in the core. This interrupt does not guarantee that a NAK handshake is sent on the USB. A STALL bit takes priority over a NAK bit. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
5	intknepmis	IN Token Received with EP Mismatch (INTknEPMis) Applies to non-periodic IN endpoints only. Indicates that the data in the top of the non-periodic TxFIFO belongs to an endpoint other than the one For which the IN token was received. This interrupt is asserted on the endpoint For which the IN token was received. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
4	intkntxfemp	IN Token Received When TxFIFO is Empty (INTknTXFEmp) Applies to non-periodic IN endpoints only. Indicates that an IN token was received when the associated TxFIFO (periodic/non-periodic) was empty. This interrupt is asserted on the endpoint For which the IN token was received. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
3	timeout	Timeout Condition (TimeOUT) In shared TX FIFO mode, applies to non-isochronous IN endpoints only. In dedicated FIFO mode, applies only to Control IN endpoints. In Scatter/Gather DMA mode, the TimeOUT interrupt is not asserted. Indicates that the core has detected a timeout condition on the USB For the last IN token on this endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
2	ahberr	AHB Error (AHBErr) Applies to IN and OUT endpoints. This is generated only in Internal DMA mode when there is an AHB error during an AHB read/write. The application can read the corresponding endpoint DMA address register to get the error address. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
1	epdisbld	Endpoint Disabled Interrupt (EPDisbld) Applies to IN and OUT endpoints. This bit indicates that the endpoint is disabled per the application's request. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
0	xfercompl	Transfer Completed Interrupt (XferCompl) Applies to IN and OUT endpoints. When Scatter/Gather DMA mode is enabled - For IN endpoint this field indicates that the requested data from the descriptor is moved from external system memory to internal FIFO. - For OUT endpoint this field indicates that the requested data from the internal FIFO is moved to external system memory. This interrupt is generated only when the corresponding endpoint descriptor is closed, and the IOC bit For the corresponding descriptor is Set. When Scatter/Gather DMA mode is disabled, this field indicates that the programmed transfer is complete on the AHB as well as on the USB, For this endpoint. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

dieptsiz15

Device IN Endpoint 15 Transfer Size Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00AF0
i_usbotg_1_devgrp	0xFFB40800	0xFFB40AF0

Offset: 0x2F0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	mc RW 0x0		pktcnt RW 0x0										xfersize RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
xfersize RW 0x0															

dieptsiz15 Fields

Bit	Name	Description	Access	Reset
30:29	mc	Applies to IN endpoints only. For periodic IN endpoints, this field indicates the number of packets that must be transmitted per microframe on the USB. The core uses this field to calculate the data PID for isochronous IN endpoints. 2'b01: 1 packet 2'b10: 2 packets 2'b11: 3 packets For non-periodic IN endpoints, this field is valid only in Internal DMA mode. It specifies the number of packets the core must fetchfor an IN endpoint before it switches to the endpoint pointed to by the Next Endpoint field of the Device Endpoint-n Control register (DIEPCTLn.NextEp) **Value** **Description** 0x1 PACKETONE 0x2 PACKETTWO 0x3 PACKETTHREE	RW	0x0

Bit	Name	Description	Access	Reset
28:19	pktcnt	Packet Count (PktCnt) Indicates the total number of USB packets that constitute the Transfer Size amount of data For endpoint 0. This field is decremented every time a packet (maximum size or short packet) is read from the TxFIFO.	RW	0x0
18:0	xfersize	Transfer Size (XferSize) Indicates the transfer size in bytes For endpoint 0. The core interrupts the application only after it has exhausted the transfer size amount of data. The transfer size can be Set to the maximum packet size of the endpoint, to be interrupted at the end of each packet. The core decrements this field every time a packet from the external memory is written to the TxFIFO.	RW	0x0

diepdma15

Device IN Endpoint 15 DMA Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00AF4
i_usbotg_1_devgrp	0xFFB40800	0xFFB40AF4

Offset: 0x2F4

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepdma15 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
diepdma15 RW 0x0															

diepdma15 Fields

Bit	Name	Description	Access	Reset
31:0	diepdma15	Holds the start address of the external memory for storing or fetching endpoint data. Note: For control endpoints, this field stores control OUT data packets as well as SETUP transaction data packets. When more than three SETUP packets are received back-to-back, the SETUP data packet in the memory is overwritten. This register is incremented on every AHB transaction. The application can give only a DWORD-aligned address. When Scatter/Gather DMA mode is not enabled, the application programs the start address value in this field. When Scatter/Gather DMA mode is enabled, this field indicates the base pointer for the descriptor list.	RW	0x0

dtxfsts15

Device IN Endpoint Transmit FIFO Status Register 15

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00AF8
i_usbotg_1_devgrp	0xFFB40800	0xFFB40AF8

Offset: 0x2F8

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ineptxfspcavail RO 0x2000															

dtxfsts15 Fields

Bit	Name	Description	Access	Reset
15:0	ineptxfspcavail	IN Endpoint TxFIFO Space Avail (INEPTxFSpcAvail) Indicates the amount of free space available in the Endpoint TxFIFO. Values are in terms of 32-bit words. 16'h0: Endpoint TxFIFO is full 16'h1: 1 word available 16'h2: 2 words available 16'hn: n words available (where 0 n 32,768) 16'h8000: 32,768 words available Others: Reserved	RO	0x2000

diepdmab15

Device IN Endpoint 15 Buffer Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00AFC
i_usbotg_1_devgrp	0xFFB40800	0xFFB40AFC

Offset: 0x2FC

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
diepdmab15 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
diepdmab15 RO 0x0															

diepdmab15 Fields

Bit	Name	Description	Access	Reset
31:0	diepdmab15	Holds the current buffer address.This register is updated as and when the data transfer for the corresponding end point is in progress. This register is present only in Scatter/Gather DMA mode. Otherwise this field is reserved.	RO	0x0

doepctl0

Device Control OUT Endpoint 0 Control Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00B00
i_usbotg_1_devgrp	0xFFB40800	0xFFB40B00

Offset: 0x300

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
epena RW 0x0	epdis RO 0x0	Reserved		snak WO 0x0	cnak WO 0x0	Reserved				stall RW 0x0	snp RW 0x0	eptype RO 0x0		naksts RO 0x0	Reserved
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
usbactep RO 0x1	Reserved													mps RO 0x0	

doepctl0 Fields

Bit	Name	Description	Access	Reset
31	epena	Endpoint Enable (EPEna) When Scatter/Gather DMA mode is enabled, For OUT endpoints this bit indicates that the descriptor structure and data buffer to receive data is setup. When Scatter/Gather DMA mode is disabled(such as For buffer-pointer based DMA mode)this bit indicates that the application has allocated the memory to start receiving data from the USB. The core clears this bit before setting any of the following interrupts on this endpoint: SETUP Phase Done Endpoint Disabled Transfer Completed Note: In DMA mode, this bit must be Set For the core to transfer SETUP data packets into memory. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
30	epdis	Endpoint Disable (EPDis) The application cannot disable control OUT endpoint 0. **Value** / **Description** 0x0 — INACTIVE	RO	0x0
27	snak	Set NAK (SNAK) A write to this bit sets the NAK bit For the endpoint. Using this bit, the application can control the transmission of NAK handshakes on an endpoint. The core can also Set bit on a Transfer Completed interrupt, or after a SETUP is received on the endpoint. **Value** / **Description** 0x0 — NOSET 0x1 — SET	WO	0x0
26	cnak	Clear NAK (CNAK) A write to this bit clears the NAK bit For the endpoint. **Value** / **Description** 0x0 — NOCLEAR 0x1 — CLEAR	WO	0x0

Bit	Name	Description	Access	Reset
21	stall	STALL Handshake (Stall) The application can only Set this bit, and the core clears it, when a SETUP token is received For this endpoint. If a NAK bit or Global OUT NAK is Set along with this bit, the STALL bit takes priority. Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
20	snp	Snoop Mode (Snp) This bit configures the endpoint to Snoop mode. In Snoop mode, the core does not check the correct-ness of OUT packets before transferring them to application memory. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
19:18	eptype	Endpoint Type (EPType) Hardcoded to 2'b00 For control. **Value** **Description** 0x0 ACTIVE	RO	0x0

Bit	Name	Description	Access	Reset
17	naksts	NAK Status (NAKSts) Indicates the following: 1'b0: The core is transmitting non-NAK handshakes based on the FIFO status. 1'b1: The core is transmitting NAK handshakes on this endpoint. When either the application or the core sets this bit, the core stops receiving data, even If there is space in the RxFIFO to accommodate the incoming packet. Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0
15	usbactep	USB Active Endpoint (USBActEP) This bit is always Set to 1, indicating that a control endpoint 0 is always active in all configurations and interfaces. **Value** **Description** 0x1 ACTIVE	RO	0x1

Bit	Name	Description	Access	Reset
1:0	mps	Maximum Packet Size (MPS) The maximum packet size For control OUT endpoint 0 is the same as what is programmed in control IN Endpoint 0. 2'b00: 64 bytes 2'b01: 32 bytes 2'b10: 16 bytes 2'b11: 8 bytes **Value** / **Description** 0x0 BYTE64 0x1 BYTE32 0x2 BYTE16 0x3 BYTE8	RO	0x0

doepint0

Device OUT Endpoint 0 Interrupt Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00B08
i_usbotg_1_devgrp	0xFFB40800	0xFFB40B08

Offset: 0x308

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
stuppktrcvd RW 0x0	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	outpkterr RW 0x0	Reserved	back2backsetup RW 0x0	stsphsercvd RW 0x0	outtknepdis RW 0x0	setup RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0

doepint0 Fields

Bit	Name	Description	Access	Reset
15	stuppktrcvd	Setup Packet Received Applicable for Control OUT Endpoints in only in the Buffer DMA Mode Set by the DWC_otg core, this bit indicates that this buffer holds 8 bytes of setup data. There is only one Setup packet per buffer. On receiving a Setup packet, the DWC_otg core closes the buffer and disables the corresponding endpoint. The applica-tion has to re-enable the endpoint to receive any OUT data for the Control Transfer and reprogram the buffer start address. Note: Because of the above behavior, the DWC_otg core can receive any number of back to back setup packets and one buffer for every setup packet is used. 1'b0: No Setup packet received 1'b1: Setup packet received Reset: 1'b0	RW	0x0
14	nyetintrpt	NYET Interrupt (NYETIntrpt) The core generates this interrupt when a NYET response is transmitted for a non isochronous OUT endpoint. Value: 0x0 — INACTIVE Value: 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
13	nakintrpt	NAK Interrupt (NAKInterrupt) The core generates this interrupt when a NAK is transmitted or received by the device. In case of isochronous IN endpoints the interrupt gets generated when a zero length packet is transmitted due to un-availability of data in the TXFifo. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
12	bbleerr	NAK Interrupt (BbleErr) The core generates this interrupt when babble is received for the endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
11	pktdrpsts	Packet Drop Status (PktDrpSts) This bit indicates to the applica-tion that an ISOC OUT packet has been dropped. This bit does not have an associated mask bit and does not generate an interrupt. Dependency: This bit is valid in non Scatter/Gather DMA mode when periodic transfer interrupt feature is selected. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
9	bnaintr	BNA (Buffer Not Available) Interrupt (BNAIntr) This bit is valid only when Scatter/ Gather DMA mode is enabled. The core generates this interrupt when the descriptor accessed is not ready For the Core to process, such as Host busy or DMA done **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
8	outpkterr	OUT Packet Error (OutPktErr) Applies to OUT endpoints Only This interrupt is valid only when thresholding is enabled. This interrupt is asserted when the core detects an overflow or a CRC error For non-Isochronous OUT packet. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
6	back2backsetup	Back-to-Back SETUP Packets Received (Back2BackSETup) Applies to Control OUT endpoints only. This bit indicates that the core has received more than three back-to-back SETUP packets For this particular endpoint. For information about handling this interrupt, **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
5	stsphsercvd	Status Phase Received For Control Write (StsPhseRcvd) This interrupt is valid only For Control OUT endpoints and only in Scatter Gather DMA mode. This interrupt is generated only after the core has transferred all the data that the host has sent during the data phase of a control write transfer, to the system memory buffer. The interrupt indicates to the application that the host has switched from data phase to the status phase of a Control Write transfer. The application can use this interrupt to ACK or STALL the Status phase, after it has decoded the data phase. This is applicable only in Case of Scatter Gather DMA mode. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
4	outtknepdis	OUT Token Received When Endpoint Disabled (OUTTknEPdis) Applies only to control OUT endpoints. Indicates that an OUT token was received when the endpoint was not yet enabled. This interrupt is asserted on the endpoint For which the OUT token was received. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
3	setup	SETUP Phase Done (SetUp) Applies to control OUT endpoints only. Indicates that the SETUP phase For the control endpoint is complete and no more back-to-back SETUP packets were received For the current control transfer. On this interrupt, the application can decode the received SETUP data packet. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
2	ahberr	AHB Error (AHBErr) Applies to IN and OUT endpoints. This is generated only in Internal DMA mode when there is an AHB error during an AHB read/write. The application can read the corresponding endpoint DMA address register to get the error address. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
1	epdisbld	Endpoint Disabled Interrupt (EPDisbld) Applies to IN and OUT endpoints. This bit indicates that the endpoint is disabled per the application's request. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
0	xfercompl	Transfer Completed Interrupt (XferCompl) Applies to IN and OUT endpoints. When Scatter/Gather DMA mode is enabled - For IN endpoint this field indicates that the requested data from the descriptor is moved from external system memory to internal FIFO. - For OUT endpoint this field indicates that the requested data from the internal FIFO is moved to external system memory. This interrupt is generated only when the corresponding endpoint descriptor is closed, and the IOC bit For the corresponding descriptor is Set. When Scatter/Gather DMA mode is disabled, this field indicates that the programmed transfer is complete on the AHB as well as on the USB, For this endpoint. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

doeptsiz0

Device OUT Endpoint 0 Transfer Size Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00B10
i_usbotg_1_devgrp	0xFFB40800	0xFFB40B10

Offset: 0x310

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	supcnt RW 0x0		Reserved									pktcnt RW 0x0	Reserved		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									xfersize RW 0x0						

doeptsiz0 Fields

Bit	Name	Description	Access	Reset
30:29	supcnt	SETUP Packet Count (SUPCnt) This field specifies the number of back-to-back SETUP data packets the endpoint can receive. 2'b01: 1 packet 2'b10: 2 packets 2'b11: 3 packets **Value** **Description** 0x1 ONEPACKET 0x2 TWOPACKET 0x3 THREEPACKET	RW	0x0
19	pktcnt	Packet Count (PktCnt) This field is decremented to zero after a packet is written into the RxFIFO.	RW	0x0

Bit	Name	Description	Access	Reset
6:0	xfersize	Transfer Size (XferSize) Indicates the transfer size in bytes For endpoint 0. The core interrupts the application only after it has exhausted the transfer size amount of data. The transfer size can be Set to the maximum packet size of the endpoint, to be interrupted at the end of each packet. The core decrements this field every time a packet is read from the RxFIFO and written to the external memory.	RW	0x0

doepdma0

Device OUT Endpoint 0 DMA Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00B14
i_usbotg_1_devgrp	0xFFB40800	0xFFB40B14

Offset: 0x314

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdma0 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
doepdma0 RW 0x0															

doepdma0 Fields

Bit	Name	Description	Access	Reset
31:0	doepdma0	Holds the start address of the external memory for storing or fetching endpoint data. Note: For control endpoints, this field stores control OUT data packets as well as SETUP transaction data packets. When more than three SETUP packets are received back-to-back, the SETUP data packet in the memory is overwritten. This register is incremented on every AHB transaction. The application can give only a DWORD-aligned address. When Scatter/Gather DMA mode is not enabled, the application programs the start address value in this field. When Scatter/Gather DMA mode is enabled, this field indicates the base pointer for the descriptor list.	RW	0x0

doepdmab0

Device OUT Endpoint 16 Buffer Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00B1C
i_usbotg_1_devgrp	0xFFB40800	0xFFB40B1C

Offset: 0x31C

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdmab0 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
doepdmab0 RO 0x0															

doepdmab0 Fields

Bit	Name	Description	Access	Reset
31:0	doepdmab0	Holds the current buffer address.This register is updated as and when the data transfer for the corresponding end point is in progress. This register is present only in Scatter/Gather DMA mode. Otherwise this field is reserved.	RO	0x0

doepctl1

Device Control OUT Endpoint 1 Control Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00B20
i_usbotg_1_devgrp	0xFFB40800	0xFFB40B20

Offset: 0x320

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	Reserved				stall RW 0x0	snp RW 0x0	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
usbactep RW 0x0	Reserved				mps RW 0x0										

doepctl1 Fields

Bit	Name	Description	Access	Reset
31	epena	Endpoint Enable (EPEna) Applies to IN and OUT endpoints. When Scatter/Gather DMA mode is enabled, For IN endpoints this bit indicates that the descriptor structure and data buffer with data ready to transmit is setup. For OUT endpoint it indicates that the descriptor structure and data buffer to receive data is setup. When Scatter/Gather DMA mode is enabledsuch as for buffer-pointer based DMA mode: - For IN endpoints, this bit indicates that data is ready to be transmitted on the endpoint. - For OUT endpoints, this bit indicates that the application has allocated the memory to start receiving data from the USB. - The core clears this bit before setting any of the following interrupts on this endpoint: SETUP Phase Done Endpoint Disabled Transfer Completed Note: For control endpoints in DMA mode, this bit must be set to be able to transfer SETUP data packets in memory. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
30	epdis	Endpoint Disable (EPDis) Applies to IN and OUT endpoints. The application sets this bit to stop transmitting/receiving data on an endpoint, even before the transfer for that endpoint is complete. The application must wait for the Endpoint Disabled interrupt before treating the endpoint as disabled. The core clears this bit before setting the Endpoint Disabled interrupt. The application must set this bit only if Endpoint Enable is already set for this endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
29	setd1pid	Set DATA1 PID (SetD1PID) Applies to interrupt/bulk IN and OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA1. This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. Set Odd (micro)frame (SetOddFr) Applies to isochronous IN and OUT endpoints only. Writing to this field sets the Even/Odd (micro)frame (EO_FrNum) field to odd (micro)frame. This field is not applicable for Scatter/Gather DMA mode. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	WO	0x0

Bit	Name	Description	Access	Reset
28	setd0pid	Set DATA0 PID (SetD0PID) Applies to interrupt/bulk IN and OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA0. This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. 1'b0 WO In non-Scatter/Gather DMA mode: Set Even (micro)frame (SetEvenFr) Applies to isochronous IN and OUT endpoints only. Writing to this field sets the Even/Odd (micro)frame (EO_FrNum) field to even (micro) frame. When Scatter/Gather DMA mode is enabled, this field is reserved. The frame number in which to send data is in the transmit descriptor structure. The frame in which to receive data is updated in receive descriptor structure. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	WO	0x0
27	snak	Set NAK (SNAK) A write to this bit sets the NAK bit For the endpoint. Using this bit, the application can control the transmission of NAK handshakes on an endpoint. The core can also Set this bit For an endpoint after a SETUP packet is received on that endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	WO	0x0

Bit	Name	Description	Access	Reset
26	cnak	Clear NAK (CNAK) A write to this bit clears the NAK bit For the endpoint. Value: 0x0 — Description: INACTIVE Value: 0x1 — Description: ACTIVE	WO	0x0
21	stall	STALL Handshake (Stall) Applies to non-control, non-isochronous IN and OUT endpoints only. The application sets this bit to stall all tokens from the USB host to this endpoint. If a NAK bit, Global Non-periodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority. Only the application can clear this bit, never the core. 1'b0 R_W Applies to control endpoints only. The application can only set this bit, and the core clears it, when a SETUP token is received for this endpoint. If a NAK bit, Global Non-periodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority. Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake. Value: 0x0 — Description: INACTIVE Value: 0x1 — Description: ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
20	snp	Snoop Mode (Snp) Applies to OUT endpoints only. This bit configures the endpoint to Snoop mode. In Snoop mode, the core does not check the correctness of OUT packets before transferring them to application memory. **Value** **Description** 0x0 DISABLE 0x1 ENABLE	RW	0x0
19:18	eptype	Endpoint Type (EPType) This is the transfer type supported by this logical endpoint. 2'b00: Control 2'b01: Isochronous 2'b10: Bulk 2'b11: Interrupt **Value** **Description** 0x0 CONTROL 0x1 ISOCHRONOUS 0x2 BULK 0x3 INTERRUP	RW	0x0

Bit	Name	Description	Access	Reset
17	naksts	NAK Status (NAKSts) Indicates the following: 1'b0: The core is transmitting non-NAK handshakes based on the FIFO status. 1'b1: The core is transmitting NAK handshakes on this endpoint. When either the application or the core sets this bit: The core stops receiving any data on an OUT endpoint, even if there is space in the RxFIFO to accommodate the incoming packet. For non-isochronous IN endpoints: The core stops transmitting any data on an IN endpoint, even if there data is available in the TxFIFO. For isochronous IN endpoints: The core sends out a zero-length data packet, even if there data is available in the TxFIFO. Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake. **Value** — **Description** 0x0 — NONNAK 0x1 — NAK	RO	0x0

Bit	Name	Description	Access	Reset
16	dpid	Endpoint Data PID (DPID) Applies to interrupt/bulk IN and OUT endpoints only. Contains the PID of the packet to be received or transmitted on this endpoint. The application must program the PID of the first packet to be received or transmitted on this endpoint, after the endpoint is activated. The applications use the SetD1PID and SetD0PID fields of this register to program either DATA0 or DATA1 PID. 1'b0: DATA0 1'b1: DATA1 This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. 1'b0 RO Even/Odd (Micro)Frame (EO_FrNum) In non-Scatter/Gather DMA mode: Applies to isochronous IN and OUT endpoints only. Indicates the (micro)frame number in which the core transmits/receives isochronous data for this endpoint. The application must program the even/odd (micro) frame number in which it intends to transmit/receive isochronous data for this endpoint using the SetEvnFr and SetOddFr fields in this register. 1'b0: Even (micro)frame 1'b1: Odd (micro)frame When Scatter/Gather DMA mode is enabled, this field is reserved. The frame number in which to send data is provided in the transmit descriptor structure. The frame in which data is received is updated in receive descriptor structure. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RO	0x0

Bit	Name	Description	Access	Reset
15	usbactep	USB Active Endpoint (USBActEP) Indicates whether this endpoint is active in the current configuration and interface. The core clears this bit for all endpoints (other than EP 0) after detecting a USB reset. After receiving the SetConfiguration and SetInterface commands, the application must program endpoint registers accordingly and set this bit. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
10:0	mps	Maximum Packet Size (MPS) The application must program this field with the maximum packet size for the current logical endpoint. This value is in bytes.	RW	0x0

doepint1

Device OUT Endpoint 1 Interrupt Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00B28
i_usbotg_1_devgrp	0xFFB40800	0xFFB40B28

Offset: 0x328

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
stuppktrcvd RW 0x0	nyetintrpt RW 0x0	nakintrpt RW 0x0	babblerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	outpkterr RW 0x0	Reserved	back2backsetup RW 0x0	stsphsercvd RW 0x0	outtknepdis RW 0x0	setup RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0

doepint1 Fields

Bit	Name	Description	Access	Reset
15	stuppktrcvd	Setup Packet Received Applicable for Control OUT Endpoints in only in the Buffer DMA Mode Set by the DWC_otg core, this bit indicates that this buffer holds 8 bytes of setup data. There is only one Setup packet per buffer. On receiving a Setup packet, the DWC_otg core closes the buffer and disables the corresponding endpoint. The applica-tion has to re-enable the endpoint to receive any OUT data for the Control Transfer and reprogram the buffer start address. Note: Because of the above behavior, the DWC_otg core can receive any number of back to back setup packets and one buffer for every setup packet is used. 1'b0: No Setup packet received 1'b1: Setup packet received Reset: 1’b0	RW	0x0

Bit	Name	Description	Access	Reset
14	nyetintrpt	NYET Interrupt (NYETIntrpt) The core generates this interrupt when a NYET response is transmitted for a non isochronous OUT endpoint. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
13	nakintrpt	NAK Interrupt (NAKInterrupt) The core generates this interrupt when a NAK is transmitted or received by the device. In case of isochronous IN endpoints the interrupt gets generated when a zero length packet is transmitted due to un-availability of data in the TXFifo. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
12	bbleerr	NAK Interrupt (BbleErr) The core generates this interrupt when babble is received for the endpoint. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
11	pktdrpsts	Packet Drop Status (PktDrpSts) This bit indicates to the application that an ISOC OUT packet has been dropped. This bit does not have an associated mask bit and does not generate an interrupt. Dependency: This bit is valid in non Scatter/Gather DMA mode when periodic transfer interrupt feature is selected. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
9	bnaintr	BNA (Buffer Not Available) Interrupt (BNAIntr) This bit is valid only when Scatter/Gather DMA mode is enabled. The core generates this interrupt when the descriptor accessed is not ready For the Core to process, such as Host busy or DMA done **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
8	outpkterr	OUT Packet Error (OutPktErr) Applies to OUT endpoints Only This interrupt is valid only when thresholding is enabled. This interrupt is asserted when the core detects an overflow or a CRC error For non-Isochronous OUT packet. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
6	back2backsetup	Back-to-Back SETUP Packets Received (Back2BackSETup) Applies to Control OUT endpoints only. This bit indicates that the core has received more than three back-to-back SETUP packets For this particular endpoint. For information about handling this interrupt, **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
5	stsphsercvd	Status Phase Received For Control Write (StsPhseRcvd) This interrupt is valid only For Control OUT endpoints and only in Scatter Gather DMA mode. This interrupt is generated only after the core has transferred all the data that the host has sent during the data phase of a control write transfer, to the system memory buffer. The interrupt indicates to the application that the host has switched from data phase to the status phase of a Control Write transfer. The application can use this interrupt to ACK or STALL the Status phase, after it has decoded the data phase. This is applicable only in Case of Scatter Gather DMA mode. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
4	outtknepdis	OUT Token Received When Endpoint Disabled (OUTTknEPdis) Applies only to control OUT endpoints. Indicates that an OUT token was received when the endpoint was not yet enabled. This interrupt is asserted on the endpoint For which the OUT token was received. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
3	setup	SETUP Phase Done (SetUp) Applies to control OUT endpoints only. Indicates that the SETUP phase For the control endpoint is complete and no more back-to-back SETUP packets were received For the current control transfer. On this interrupt, the application can decode the received SETUP data packet. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
2	ahberr	AHB Error (AHBErr) Applies to IN and OUT endpoints. This is generated only in Internal DMA mode when there is an AHB error during an AHB read/write. The application can read the corresponding endpoint DMA address register to get the error address. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
1	epdisbld	Endpoint Disabled Interrupt (EPDisbld) Applies to IN and OUT endpoints. This bit indicates that the endpoint is disabled per the application's request. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
0	xfercompl	Transfer Completed Interrupt (XferCompl) Applies to IN and OUT endpoints. When Scatter/Gather DMA mode is enabled - For IN endpoint this field indicates that the requested data from the descriptor is moved from external system memory to internal FIFO. - For OUT endpoint this field indicates that the requested data from the internal FIFO is moved to external system memory. This interrupt is generated only when the corresponding endpoint descriptor is closed, and the IOC bit For the corresponding descriptor is Set. When Scatter/Gather DMA mode is disabled, this field indicates that the programmed transfer is complete on the AHB as well as on the USB, For this endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

doeptsiz1

Device OUT Endpoint 1 Transfer Size Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00B30
i_usbotg_1_devgrp	0xFFB40800	0xFFB40B30

Offset: 0x330

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	rxdpid RO 0x0		pktcnt RW 0x0										xfersize RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
xfersize RW 0x0															

doeptsiz1 Fields

Bit	Name	Description	Access	Reset
30:29	rxdpid	Applies to isochronous OUT endpoints only. This is the data PID received in the last packet for this endpoint. 2'b00: DATA0 2'b01: DATA2 2'b10: DATA1 2'b11: MDATA SETUP Packet Count (SUPCnt) Applies to control OUT Endpoints only. This field specifies the number of back-to-back SETUP data packets the endpoint can receive. 2'b01: 1 packet 2'b10: 2 packets 2'b11: 3 packets **Value** **Description** 0x0 DATA0 0x1 DATA2PACKET1 0x2 DATA1PACKET2 0x3 MDATAPACKET3	RO	0x0
28:19	pktcnt	Packet Count (PktCnt) This field is decremented to zero after a packet is written into the RxFIFO.	RW	0x0
18:0	xfersize	Transfer Size (XferSize) Indicates the transfer size in bytes For endpoint 0. The core interrupts the application only after it has exhausted the transfer size amount of data. The transfer size can be Set to the maximum packet size of the endpoint, to be interrupted at the end of each packet. The core decrements this field every time a packet is read from the RxFIFO and written to the external memory.	RW	0x0

doepdma1

Device OUT Endpoint 1 DMA Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00B34
i_usbotg_1_devgrp	0xFFB40800	0xFFB40B34

Offset: 0x334

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdma1 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
doepdma1 RW 0x0															

doepdma1 Fields

Bit	Name	Description	Access	Reset
31:0	doepdma1	Holds the start address of the external memory for storing or fetching endpoint data. Note: For control endpoints, this field stores control OUT data packets as well as SETUP transaction data packets. When more than three SETUP packets are received back-to-back, the SETUP data packet in the memory is overwritten. This register is incremented on every AHB transaction. The application can give only a DWORD-aligned address. When Scatter/Gather DMA mode is not enabled, the application programs the start address value in this field. When Scatter/Gather DMA mode is enabled, this field indicates the base pointer for the descriptor list.	RW	0x0

doepdmab1

Device OUT Endpoint 1 Buffer Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00B3C
i_usbotg_1_devgrp	0xFFB40800	0xFFB40B3C

Offset: 0x33C

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdmab1 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
doepdmab1 RO 0x0															

doepdmab1 Fields

Bit	Name	Description	Access	Reset
31:0	doepdmab1	Holds the current buffer address.This register is updated as and when the data transfer for the corresponding end point is in progress. This register is present only in Scatter/Gather DMA mode. Otherwise this field is reserved.	RO	0x0

doepctl2

Device Control OUT Endpoint 2 Control Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00B40
i_usbotg_1_devgrp	0xFFB40800	0xFFB40B40

Offset: 0x340

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	Reserved				stall RW 0x0	snp RW 0x0	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
usbactep RW 0x0	Reserved				mps RW 0x0										

doepctl2 Fields

Bit	Name	Description	Access	Reset
31	epena	Endpoint Enable (EPEna) Applies to IN and OUT endpoints. When Scatter/Gather DMA mode is enabled, For IN endpoints this bit indicates that the descriptor structure and data buffer with data ready to transmit is setup. For OUT endpoint it indicates that the descriptor structure and data buffer to receive data is setup. When Scatter/Gather DMA mode is enabledsuch as for buffer-pointer based DMA mode: - For IN endpoints, this bit indicates that data is ready to be transmitted on the endpoint. - For OUT endpoints, this bit indicates that the application has allocated the memory to start receiving data from the USB. - The core clears this bit before setting any of the following interrupts on this endpoint: SETUP Phase Done Endpoint Disabled Transfer Completed Note: For control endpoints in DMA mode, this bit must be set to be able to transfer SETUP data packets in memory. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
30	epdis	Endpoint Disable (EPDis) Applies to IN and OUT endpoints. The application sets this bit to stop transmitting/receiving data on an endpoint, even before the transfer for that endpoint is complete. The application must wait for the Endpoint Disabled interrupt before treating the endpoint as disabled. The core clears this bit before setting the Endpoint Disabled interrupt. The application must set this bit only if Endpoint Enable is already set for this endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
29	setd1pid	Set DATA1 PID (SetD1PID) Applies to interrupt/bulk IN and OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA1. This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. Set Odd (micro)frame (SetOddFr) Applies to isochronous IN and OUT endpoints only. Writing to this field sets the Even/Odd (micro)frame (EO_FrNum) field to odd (micro)frame. This field is not applicable for Scatter/Gather DMA mode. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	WO	0x0

Bit	Name	Description	Access	Reset
28	setd0pid	Set DATA0 PID (SetD0PID) Applies to interrupt/bulk IN and OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA0. This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. 1'b0 WO In non-Scatter/Gather DMA mode: Set Even (micro)frame (SetEvenFr) Applies to isochronous IN and OUT endpoints only. Writing to this field sets the Even/Odd (micro)frame (EO_FrNum) field to even (micro) frame. When Scatter/Gather DMA mode is enabled, this field is reserved. The frame number in which to send data is in the transmit descriptor structure. The frame in which to receive data is updated in receive descriptor structure. **Value: Description** 0x0: DISABLED 0x1: ENABLED	WO	0x0
27	snak	Set NAK (SNAK) A write to this bit sets the NAK bit For the endpoint. Using this bit, the application can control the transmission of NAK handshakes on an endpoint. The core can also Set this bit For an endpoint after a SETUP packet is received on that endpoint. **Value: Description** 0x0: INACTIVE 0x1: ACTIVE	WO	0x0

Bit	Name	Description	Access	Reset
26	cnak	Clear NAK (CNAK) A write to this bit clears the NAK bit For the endpoint. Value: 0x0 — INACTIVE Value: 0x1 — ACTIVE	WO	0x0
21	stall	STALL Handshake (Stall) Applies to non-control, non-isochronous IN and OUT endpoints only. The application sets this bit to stall all tokens from the USB host to this endpoint. If a NAK bit, Global Non-periodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority. Only the application can clear this bit, never the core. 1'b0 R_W Applies to control endpoints only. The application can only set this bit, and the core clears it, when a SETUP token is received for this endpoint. If a NAK bit, Global Non-periodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority. Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake. Value: 0x0 — INACTIVE Value: 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
20	snp	Snoop Mode (Snp) Applies to OUT endpoints only. This bit configures the endpoint to Snoop mode. In Snoop mode, the core does not check the correctness of OUT packets before transferring them to application memory. **Value** — **Description** 0x0 — DISABLE 0x1 — ENABLE	RW	0x0
19:18	eptype	Endpoint Type (EPType) This is the transfer type supported by this logical endpoint. 2'b00: Control 2'b01: Isochronous 2'b10: Bulk 2'b11: Interrupt **Value** — **Description** 0x0 — CONTROL 0x1 — ISOCHRONOUS 0x2 — BULK 0x3 — INTERRUP	RW	0x0

Bit	Name	Description	Access	Reset
17	naksts	NAK Status (NAKSts) Indicates the following: 1'b0: The core is transmitting non-NAK handshakes based on the FIFO status. 1'b1: The core is transmitting NAK handshakes on this endpoint. When either the application or the core sets this bit: The core stops receiving any data on an OUT endpoint, even if there is space in the RxFIFO to accommodate the incoming packet. For non-isochronous IN endpoints: The core stops transmitting any data on an IN endpoint, even if there data is available in the TxFIFO. For isochronous IN endpoints: The core sends out a zero-length data packet, even if there data is available in the TxFIFO. Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake. **Value** — **Description** 0x0 — NONNAK 0x1 — NAK	RO	0x0

Bit	Name	Description	Access	Reset
16	dpid	Endpoint Data PID (DPID) Applies to interrupt/bulk IN and OUT endpoints only. Contains the PID of the packet to be received or transmitted on this endpoint. The application must program the PID of the first packet to be received or transmitted on this endpoint, after the endpoint is activated. The applications use the SetD1PID and SetD0PID fields of this register to program either DATA0 or DATA1 PID. 1'b0: DATA0 1'b1: DATA1 This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. 1'b0 RO Even/Odd (Micro)Frame (EO_FrNum) In non-Scatter/Gather DMA mode: Applies to isochronous IN and OUT endpoints only. Indicates the (micro)frame number in which the core transmits/ receives isochronous data for this endpoint. The application must program the even/odd (micro) frame number in which it intends to transmit/receive isochronous data for this endpoint using the SetEvnFr and SetOddFr fields in this register. 1'b0: Even (micro)frame 1'b1: Odd (micro)frame When Scatter/Gather DMA mode is enabled, this field is reserved. The frame number in which to send data is provided in the transmit descriptor structure. The frame in which data is received is updated in receive descriptor structure. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RO	0x0

Bit	Name	Description	Access	Reset
15	usbactep	USB Active Endpoint (USBActEP) Indicates whether this endpoint is active in the current configuration and interface. The core clears this bit for all endpoints (other than EP 0) after detecting a USB reset. After receiving the SetConfiguration and SetInterface commands, the applica-tion must program endpoint registers accordingly and set this bit. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
10:0	mps	Maximum Packet Size (MPS) The application must program this field with the maximum packet size for the current logical endpoint. This value is in bytes.	RW	0x0

doepint2

Device OUT Endpoint 2 Interrupt Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00B48
i_usbotg_1_devgrp	0xFFB40800	0xFFB40B48

Offset: 0x348

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
stuppktrcvd RW 0x0	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	outpkterr RW 0x0	Reserved	back2backsetup RW 0x0	stsphsercvd RW 0x0	outtknepdis RW 0x0	setup RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0

doepint2 Fields

Bit	Name	Description	Access	Reset
15	stuppktrcvd	Setup Packet Received Applicable for Control OUT Endpoints in only in the Buffer DMA Mode Set by the DWC_otg core, this bit indicates that this buffer holds 8 bytes of setup data. There is only one Setup packet per buffer. On receiving a Setup packet, the DWC_otg core closes the buffer and disables the corresponding endpoint. The applica-tion has to re-enable the endpoint to receive any OUT data for the Control Transfer and reprogram the buffer start address. Note: Because of the above behavior, the DWC_otg core can receive any number of back to back setup packets and one buffer for every setup packet is used. 1'b0: No Setup packet received 1'b1: Setup packet received Reset: 1’b0	RW	0x0

Bit	Name	Description	Access	Reset
14	nyetintrpt	NYET Interrupt (NYETIntrpt) The core generates this interrupt when a NYET response is transmitted for a non isochronous OUT endpoint. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
13	nakintrpt	NAK Interrupt (NAKInterrupt) The core generates this interrupt when a NAK is transmitted or received by the device. In case of isochronous IN endpoints the interrupt gets generated when a zero length packet is transmitted due to un-availability of data in the TXFifo. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
12	bbleerr	NAK Interrupt (BbleErr) The core generates this interrupt when babble is received for the endpoint. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
11	pktdrpsts	Packet Drop Status (PktDrpSts) This bit indicates to the application that an ISOC OUT packet has been dropped. This bit does not have an associated mask bit and does not generate an interrupt. Dependency: This bit is valid in non Scatter/Gather DMA mode when periodic transfer interrupt feature is selected. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
9	bnaintr	BNA (Buffer Not Available) Interrupt (BNAIntr) This bit is valid only when Scatter/Gather DMA mode is enabled. The core generates this interrupt when the descriptor accessed is not ready For the Core to process, such as Host busy or DMA done **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
8	outpkterr	OUT Packet Error (OutPktErr) Applies to OUT endpoints Only This interrupt is valid only when thresholding is enabled. This interrupt is asserted when the core detects an overflow or a CRC error For non-Isochronous OUT packet. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
6	back2backsetup	Back-to-Back SETUP Packets Received (Back2BackSETup) Applies to Control OUT endpoints only. This bit indicates that the core has received more than three back-to-back SETUP packets For this particular endpoint. For information about handling this interrupt, **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
5	stsphsercvd	Status Phase Received For Control Write (StsPhseRcvd) This interrupt is valid only For Control OUT endpoints and only in Scatter Gather DMA mode. This interrupt is generated only after the core has transferred all the data that the host has sent during the data phase of a control write transfer, to the system memory buffer. The interrupt indicates to the application that the host has switched from data phase to the status phase of a Control Write transfer. The application can use this interrupt to ACK or STALL the Status phase, after it has decoded the data phase. This is applicable only in Case of Scatter Gather DMA mode. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
4	outtknepdis	OUT Token Received When Endpoint Disabled (OUTTknEPdis) Applies only to control OUT endpoints. Indicates that an OUT token was received when the endpoint was not yet enabled. This interrupt is asserted on the endpoint For which the OUT token was received. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
3	setup	SETUP Phase Done (SetUp) Applies to control OUT endpoints only. Indicates that the SETUP phase For the control endpoint is complete and no more back-to-back SETUP packets were received For the current control transfer. On this interrupt, the application can decode the received SETUP data packet. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
2	ahberr	AHB Error (AHBErr) Applies to IN and OUT endpoints. This is generated only in Internal DMA mode when there is an AHB error during an AHB read/write. The application can read the corresponding endpoint DMA address register to get the error address. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
1	epdisbld	Endpoint Disabled Interrupt (EPDisbld) Applies to IN and OUT endpoints. This bit indicates that the endpoint is disabled per the application's request. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
0	xfercompl	Transfer Completed Interrupt (XferCompl) Applies to IN and OUT endpoints. When Scatter/Gather DMA mode is enabled - For IN endpoint this field indicates that the requested data from the descriptor is moved from external system memory to internal FIFO. - For OUT endpoint this field indicates that the requested data from the internal FIFO is moved to external system memory. This interrupt is generated only when the corresponding endpoint descriptor is closed, and the IOC bit For the corresponding descriptor is Set. When Scatter/Gather DMA mode is disabled, this field indicates that the programmed transfer is complete on the AHB as well as on the USB, For this endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

doeptsiz2

Device OUT Endpoint 2 Transfer Size Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00B50
i_usbotg_1_devgrp	0xFFB40800	0xFFB40B50

Offset: 0x350

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	rxdpid RO 0x0		pktcnt RW 0x0										xfersize RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
xfersize RW 0x0															

doeptsiz2 Fields

Bit	Name	Description	Access	Reset
30:29	rxdpid	Applies to isochronous OUT endpoints only. This is the data PID received in the last packet for this endpoint. 2'b00: DATA0 2'b01: DATA2 2'b10: DATA1 2'b11: MDATA SETUP Packet Count (SUPCnt) Applies to control OUT Endpoints only. This field specifies the number of back-to-back SETUP data packets the endpoint can receive. 2'b01: 1 packet 2'b10: 2 packets 2'b11: 3 packets **Value / Description** 0x0 DATA0 0x1 DATA2PACKET1 0x2 DATA1PACKET2 0x3 MDATAPACKET3	RO	0x0
28:19	pktcnt	Packet Count (PktCnt) This field is decremented to zero after a packet is written into the RxFIFO.	RW	0x0
18:0	xfersize	Transfer Size (XferSize) Indicates the transfer size in bytes For endpoint 0. The core interrupts the application only after it has exhausted the transfer size amount of data. The transfer size can be Set to the maximum packet size of the endpoint, to be interrupted at the end of each packet. The core decrements this field every time a packet is read from the RxFIFO and written to the external memory.	RW	0x0

doepdma2

Device OUT Endpoint 2 DMA Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00B54
i_usbotg_1_devgrp	0xFFB40800	0xFFB40B54

Offset: 0x354

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdma2 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
doepdma2 RW 0x0															

doepdma2 Fields

Bit	Name	Description	Access	Reset
31:0	doepdma2	Holds the start address of the external memory for storing or fetching endpoint data. Note: For control endpoints, this field stores control OUT data packets as well as SETUP transaction data packets. When more than three SETUP packets are received back-to-back, the SETUP data packet in the memory is overwritten. This register is incremented on every AHB transaction. The application can give only a DWORD-aligned address. When Scatter/Gather DMA mode is not enabled, the application programs the start address value in this field. When Scatter/Gather DMA mode is enabled, this field indicates the base pointer for the descriptor list.	RW	0x0

doepdmab2

Device OUT Endpoint 2 Buffer Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00B5C
i_usbotg_1_devgrp	0xFFB40800	0xFFB40B5C

Offset: 0x35C

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdmab2 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
doepdmab2 RO 0x0															

doepdmab2 Fields

Bit	Name	Description	Access	Reset
31:0	doepdmab2	Holds the current buffer address.This register is updated as and when the data transfer for the corresponding end point is in progress. This register is present only in Scatter/Gather DMA mode. Otherwise this field is reserved.	RO	0x0

doepctl3

Device Control OUT Endpoint 3 Control Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00B60
i_usbotg_1_devgrp	0xFFB40800	0xFFB40B60

Offset: 0x360

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	Reserved				stall RW 0x0	snp RW 0x0	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
usbactep RW 0x0	Reserved				mps RW 0x0										

doepctl3 Fields

Bit	Name	Description	Access	Reset
31	epena	Endpoint Enable (EPEna) Applies to IN and OUT endpoints. When Scatter/Gather DMA mode is enabled, For IN endpoints this bit indicates that the descriptor structure and data buffer with data ready to transmit is setup. For OUT endpoint it indicates that the descriptor structure and data buffer to receive data is setup. When Scatter/Gather DMA mode is enabledsuch as for buffer-pointer based DMA mode: - For IN endpoints, this bit indicates that data is ready to be transmitted on the endpoint. - For OUT endpoints, this bit indicates that the application has allocated the memory to start receiving data from the USB. - The core clears this bit before setting any of the following interrupts on this endpoint: SETUP Phase Done Endpoint Disabled Transfer Completed Note: For control endpoints in DMA mode, this bit must be set to be able to transfer SETUP data packets in memory. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
30	epdis	Endpoint Disable (EPDis) Applies to IN and OUT endpoints. The application sets this bit to stop transmitting/receiving data on an endpoint, even before the transfer for that endpoint is complete. The application must wait for the Endpoint Disabled interrupt before treating the endpoint as disabled. The core clears this bit before setting the Endpoint Disabled interrupt. The application must set this bit only if Endpoint Enable is already set for this endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
29	setd1pid	Set DATA1 PID (SetD1PID) Applies to interrupt/bulk IN and OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA1. This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. Set Odd (micro)frame (SetOddFr) Applies to isochronous IN and OUT endpoints only. Writing to this field sets the Even/Odd (micro)frame (EO_FrNum) field to odd (micro)frame. This field is not applicable for Scatter/Gather DMA mode. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	WO	0x0

Bit	Name	Description	Access	Reset
28	setd0pid	Set DATA0 PID (SetD0PID) Applies to interrupt/bulk IN and OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA0. This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. 1'b0 WO In non-Scatter/Gather DMA mode: Set Even (micro)frame (SetEvenFr) Applies to isochronous IN and OUT endpoints only. Writing to this field sets the Even/Odd (micro)frame (EO_FrNum) field to even (micro) frame. When Scatter/Gather DMA mode is enabled, this field is reserved. The frame number in which to send data is in the transmit descriptor structure. The frame in which to receive data is updated in receive descriptor structure. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	WO	0x0
27	snak	Set NAK (SNAK) A write to this bit sets the NAK bit For the endpoint. Using this bit, the application can control the transmission of NAK handshakes on an endpoint. The core can also Set this bit For an endpoint after a SETUP packet is received on that endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	WO	0x0

Bit	Name	Description	Access	Reset
26	cnak	Clear NAK (CNAK) A write to this bit clears the NAK bit For the endpoint. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	WO	0x0
21	stall	STALL Handshake (Stall) Applies to non-control, non-isochronous IN and OUT endpoints only. The application sets this bit to stall all tokens from the USB host to this endpoint. If a NAK bit, Global Non-periodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority. Only the application can clear this bit, never the core. 1'b0 R_W Applies to control endpoints only. The application can only set this bit, and the core clears it, when a SETUP token is received for this endpoint. If a NAK bit, Global Non-periodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority. Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
20	snp	Snoop Mode (Snp) Applies to OUT endpoints only. This bit configures the endpoint to Snoop mode. In Snoop mode, the core does not check the correctness of OUT packets before transferring them to application memory. **Value** — **Description** 0x0 — DISABLE 0x1 — ENABLE	RW	0x0
19:18	eptype	Endpoint Type (EPType) This is the transfer type supported by this logical endpoint. 2'b00: Control 2'b01: Isochronous 2'b10: Bulk 2'b11: Interrupt **Value** — **Description** 0x0 — CONTROL 0x1 — ISOCHRONOUS 0x2 — BULK 0x3 — INTERRUP	RW	0x0

Bit	Name	Description	Access	Reset
17	naksts	NAK Status (NAKSts) Indicates the following: 1'b0: The core is transmitting non-NAK handshakes based on the FIFO status. 1'b1: The core is transmitting NAK handshakes on this endpoint. When either the application or the core sets this bit: The core stops receiving any data on an OUT endpoint, even if there is space in the RxFIFO to accommodate the incoming packet. For non-isochronous IN endpoints: The core stops transmitting any data on an IN endpoint, even if there data is available in the TxFIFO. For isochronous IN endpoints: The core sends out a zero-length data packet, even if there data is available in the TxFIFO. Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake. **Value** — **Description** 0x0 — NONNAK 0x1 — NAK	RO	0x0

Bit	Name	Description	Access	Reset
16	dpid	Endpoint Data PID (DPID) Applies to interrupt/bulk IN and OUT endpoints only. Contains the PID of the packet to be received or transmitted on this endpoint. The application must program the PID of the first packet to be received or transmitted on this endpoint, after the endpoint is activated. The applications use the SetD1PID and SetD0PID fields of this register to program either DATA0 or DATA1 PID. 1'b0: DATA0 1'b1: DATA1 This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. 1'b0 RO Even/Odd (Micro)Frame (EO_FrNum) In non-Scatter/Gather DMA mode: Applies to isochronous IN and OUT endpoints only. Indicates the (micro)frame number in which the core transmits/ receives isochronous data for this endpoint. The application must program the even/odd (micro) frame number in which it intends to transmit/receive isochronous data for this endpoint using the SetEvnFr and SetOddFr fields in this register. 1'b0: Even (micro)frame 1'b1: Odd (micro)frame When Scatter/Gather DMA mode is enabled, this field is reserved. The frame number in which to send data is provided in the transmit descriptor structure. The frame in which data is received is updated in receive descriptor structure. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RO	0x0

Bit	Name	Description	Access	Reset
15	usbactep	USB Active Endpoint (USBActEP) Indicates whether this endpoint is active in the current configuration and interface. The core clears this bit for all endpoints (other than EP 0) after detecting a USB reset. After receiving the SetConfiguration and SetInterface commands, the application must program endpoint registers accordingly and set this bit. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
10:0	mps	Maximum Packet Size (MPS) The application must program this field with the maximum packet size for the current logical endpoint. This value is in bytes.	RW	0x0

doepint3

Device OUT Endpoint 3 Interrupt Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00B68
i_usbotg_1_devgrp	0xFFB40800	0xFFB40B68

Offset: 0x368

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
stuppktrcvd RW 0x0	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	outpkterr RW 0x0	Reserved	back2backsetup RW 0x0	stsphsercvd RW 0x0	outtknepdis RW 0x0	setup RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0

doepint3 Fields

Bit	Name	Description	Access	Reset
15	stuppktrcvd	Setup Packet Received Applicable for Control OUT Endpoints in only in the Buffer DMA Mode Set by the DWC_otg core, this bit indicates that this buffer holds 8 bytes of setup data. There is only one Setup packet per buffer. On receiving a Setup packet, the DWC_otg core closes the buffer and disables the corresponding endpoint. The applica-tion has to re-enable the endpoint to receive any OUT data for the Control Transfer and reprogram the buffer start address. Note: Because of the above behavior, the DWC_otg core can receive any number of back to back setup packets and one buffer for every setup packet is used. 1'b0: No Setup packet received 1'b1: Setup packet received Reset: 1’b0	RW	0x0

Bit	Name	Description	Access	Reset
14	nyetintrpt	NYET Interrupt (NYETIntrpt) The core generates this interrupt when a NYET response is transmitted for a non isochronous OUT endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
13	nakintrpt	NAK Interrupt (NAKInterrupt) The core generates this interrupt when a NAK is transmitted or received by the device. In case of isochronous IN endpoints the interrupt gets generated when a zero length packet is transmitted due to un-availability of data in the TXFifo. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
12	bbleerr	NAK Interrupt (BbleErr) The core generates this interrupt when babble is received for the endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
11	pktdrpsts	Packet Drop Status (PktDrpSts) This bit indicates to the application that an ISOC OUT packet has been dropped. This bit does not have an associated mask bit and does not generate an interrupt. Dependency: This bit is valid in non Scatter/Gather DMA mode when periodic transfer interrupt feature is selected. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
9	bnaintr	BNA (Buffer Not Available) Interrupt (BNAIntr) This bit is valid only when Scatter/Gather DMA mode is enabled. The core generates this interrupt when the descriptor accessed is not ready For the Core to process, such as Host busy or DMA done **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
8	outpkterr	OUT Packet Error (OutPktErr) Applies to OUT endpoints Only This interrupt is valid only when thresholding is enabled. This interrupt is asserted when the core detects an overflow or a CRC error For non-Isochronous OUT packet. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
6	back2backsetup	Back-to-Back SETUP Packets Received (Back2BackSETup) Applies to Control OUT endpoints only. This bit indicates that the core has received more than three back-to-back SETUP packets For this particular endpoint. For information about handling this interrupt, **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
5	stsphsercvd	Status Phase Received For Control Write (StsPhseRcvd) This interrupt is valid only For Control OUT endpoints and only in Scatter Gather DMA mode. This interrupt is generated only after the core has transferred all the data that the host has sent during the data phase of a control write transfer, to the system memory buffer. The interrupt indicates to the application that the host has switched from data phase to the status phase of a Control Write transfer. The application can use this interrupt to ACK or STALL the Status phase, after it has decoded the data phase. This is applicable only in Case of Scatter Gather DMA mode. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
4	outtknepdis	OUT Token Received When Endpoint Disabled (OUTTknEPdis) Applies only to control OUT endpoints. Indicates that an OUT token was received when the endpoint was not yet enabled. This interrupt is asserted on the endpoint For which the OUT token was received. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
3	setup	SETUP Phase Done (SetUp) Applies to control OUT endpoints only. Indicates that the SETUP phase For the control endpoint is complete and no more back-to-back SETUP packets were received For the current control transfer. On this interrupt, the application can decode the received SETUP data packet. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
2	ahberr	AHB Error (AHBErr) Applies to IN and OUT endpoints. This is generated only in Internal DMA mode when there is an AHB error during an AHB read/write. The application can read the corresponding endpoint DMA address register to get the error address. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
1	epdisbld	Endpoint Disabled Interrupt (EPDisbld) Applies to IN and OUT endpoints. This bit indicates that the endpoint is disabled per the application's request. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
0	xfercompl	Transfer Completed Interrupt (XferCompl) Applies to IN and OUT endpoints. When Scatter/Gather DMA mode is enabled - For IN endpoint this field indicates that the requested data from the descriptor is moved from external system memory to internal FIFO. - For OUT endpoint this field indicates that the requested data from the internal FIFO is moved to external system memory. This interrupt is generated only when the corresponding endpoint descriptor is closed, and the IOC bit For the corresponding descriptor is Set. When Scatter/Gather DMA mode is disabled, this field indicates that the programmed transfer is complete on the AHB as well as on the USB, For this endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

doeptsiz3

Device OUT Endpoint 3 Transfer Size Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00B70
i_usbotg_1_devgrp	0xFFB40800	0xFFB40B70

Offset: 0x370

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	rxdpid RO 0x0		pktcnt RW 0x0										xfersize RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
xfersize RW 0x0															

doeptsiz3 Fields

Bit	Name	Description	Access	Reset
30:29	rxdpid	Applies to isochronous OUT endpoints only. This is the data PID received in the last packet for this endpoint. 2'b00: DATA0 2'b01: DATA2 2'b10: DATA1 2'b11: MDATA SETUP Packet Count (SUPCnt) Applies to control OUT Endpoints only. This field specifies the number of back-to-back SETUP data packets the endpoint can receive. 2'b01: 1 packet 2'b10: 2 packets 2'b11: 3 packets **Value** **Description** 0x0 DATA0 0x1 DATA2PACKET1 0x2 DATA1PACKET2 0x3 MDATAPACKET3	RO	0x0
28:19	pktcnt	Packet Count (PktCnt) This field is decremented to zero after a packet is written into the RxFIFO.	RW	0x0
18:0	xfersize	Transfer Size (XferSize) Indicates the transfer size in bytes For endpoint 0. The core interrupts the application only after it has exhausted the transfer size amount of data. The transfer size can be Set to the maximum packet size of the endpoint, to be interrupted at the end of each packet. The core decrements this field every time a packet is read from the RxFIFO and written to the external memory.	RW	0x0

doepdma3

Device OUT Endpoint 3 DMA Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00B74
i_usbotg_1_devgrp	0xFFB40800	0xFFB40B74

Offset: 0x374

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdma3 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
doepdma3 RW 0x0															

doepdma3 Fields

Bit	Name	Description	Access	Reset
31:0	doepdma3	Holds the start address of the external memory for storing or fetching endpoint data. Note: For control endpoints, this field stores control OUT data packets as well as SETUP transaction data packets. When more than three SETUP packets are received back-to-back, the SETUP data packet in the memory is overwritten. This register is incremented on every AHB transaction. The application can give only a DWORD-aligned address. When Scatter/Gather DMA mode is not enabled, the application programs the start address value in this field. When Scatter/Gather DMA mode is enabled, this field indicates the base pointer for the descriptor list.	RW	0x0

doepdmab3

Device OUT Endpoint 3 Buffer Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00B7C
i_usbotg_1_devgrp	0xFFB40800	0xFFB40B7C

Offset: 0x37C

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdmab3 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
doepdmab3 RO 0x0															

doepdmab3 Fields

Bit	Name	Description	Access	Reset
31:0	doepdmab3	Holds the current buffer address.This register is updated as and when the data transfer for the corresponding end point is in progress. This register is present only in Scatter/Gather DMA mode. Otherwise this field is reserved.	RO	0x0

doepctl4

Device Control OUT Endpoint 4 Control Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00B80
i_usbotg_1_devgrp	0xFFB40800	0xFFB40B80

Offset: 0x380

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	Reserved				stall RW 0x0	snp RW 0x0	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
usbactep RW 0x0	Reserved				mps RW 0x0										

doepctl4 Fields

Bit	Name	Description	Access	Reset
31	epena	Endpoint Enable (EPEna) Applies to IN and OUT endpoints. When Scatter/Gather DMA mode is enabled, For IN endpoints this bit indicates that the descriptor structure and data buffer with data ready to transmit is setup. For OUT endpoint it indicates that the descriptor structure and data buffer to receive data is setup. When Scatter/Gather DMA mode is enabledsuch as for buffer-pointer based DMA mode: - For IN endpoints, this bit indicates that data is ready to be transmitted on the endpoint. - For OUT endpoints, this bit indicates that the application has allocated the memory to start receiving data from the USB. - The core clears this bit before setting any of the following interrupts on this endpoint: SETUP Phase Done Endpoint Disabled Transfer Completed Note: For control endpoints in DMA mode, this bit must be set to be able to transfer SETUP data packets in memory. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
30	epdis	Endpoint Disable (EPDis) Applies to IN and OUT endpoints. The application sets this bit to stop transmitting/receiving data on an endpoint, even before the transfer for that endpoint is complete. The application must wait for the Endpoint Disabled interrupt before treating the endpoint as disabled. The core clears this bit before setting the Endpoint Disabled interrupt. The application must set this bit only if Endpoint Enable is already set for this endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
29	setd1pid	Set DATA1 PID (SetD1PID) Applies to interrupt/bulk IN and OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA1. This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. Set Odd (micro)frame (SetOddFr) Applies to isochronous IN and OUT endpoints only. Writing to this field sets the Even/Odd (micro)frame (EO_FrNum) field to odd (micro)frame. This field is not applicable for Scatter/Gather DMA mode. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	WO	0x0

Bit	Name	Description	Access	Reset
28	setd0pid	Set DATA0 PID (SetD0PID) Applies to interrupt/bulk IN and OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA0. This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. 1'b0 WO In non-Scatter/Gather DMA mode: Set Even (micro)frame (SetEvenFr) Applies to isochronous IN and OUT endpoints only. Writing to this field sets the Even/Odd (micro)frame (EO_FrNum) field to even (micro) frame. When Scatter/Gather DMA mode is enabled, this field is reserved. The frame number in which to send data is in the transmit descriptor structure. The frame in which to receive data is updated in receive descriptor structure. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	WO	0x0
27	snak	Set NAK (SNAK) A write to this bit sets the NAK bit For the endpoint. Using this bit, the application can control the transmission of NAK handshakes on an endpoint. The core can also Set this bit For an endpoint after a SETUP packet is received on that endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	WO	0x0

Bit	Name	Description	Access	Reset
26	cnak	Clear NAK (CNAK) A write to this bit clears the NAK bit For the endpoint. Value: 0x0 — INACTIVE Value: 0x1 — ACTIVE	WO	0x0
21	stall	STALL Handshake (Stall) Applies to non-control, non-isochronous IN and OUT endpoints only. The application sets this bit to stall all tokens from the USB host to this endpoint. If a NAK bit, Global Non-periodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority. Only the application can clear this bit, never the core. 1'b0 R_W Applies to control endpoints only. The application can only set this bit, and the core clears it, when a SETUP token is received for this endpoint. If a NAK bit, Global Non-periodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority. Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake. Value: 0x0 — INACTIVE Value: 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
20	snp	Snoop Mode (Snp) Applies to OUT endpoints only. This bit configures the endpoint to Snoop mode. In Snoop mode, the core does not check the correctness of OUT packets before transferring them to application memory. **Value** **Description** 0x0 DISABLE 0x1 ENABLE	RW	0x0
19:18	eptype	Endpoint Type (EPType) This is the transfer type supported by this logical endpoint. 2'b00: Control 2'b01: Isochronous 2'b10: Bulk 2'b11: Interrupt **Value** **Description** 0x0 CONTROL 0x1 ISOCHRONOUS 0x2 BULK 0x3 INTERRUP	RW	0x0

Bit	Name	Description	Access	Reset
17	naksts	NAK Status (NAKSts) Indicates the following: 1'b0: The core is transmitting non-NAK handshakes based on the FIFO status. 1'b1: The core is transmitting NAK handshakes on this endpoint. When either the application or the core sets this bit: The core stops receiving any data on an OUT endpoint, even if there is space in the RxFIFO to accommodate the incoming packet. For non-isochronous IN endpoints: The core stops transmitting any data on an IN endpoint, even if there data is available in the TxFIFO. For isochronous IN endpoints: The core sends out a zero-length data packet, even if there data is available in the TxFIFO. Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake. **Value** — **Description** 0x0 — NONNAK 0x1 — NAK	RO	0x0

Bit	Name	Description	Access	Reset
16	dpid	Endpoint Data PID (DPID) Applies to interrupt/bulk IN and OUT endpoints only. Contains the PID of the packet to be received or transmitted on this endpoint. The application must program the PID of the first packet to be received or transmitted on this endpoint, after the endpoint is activated. The applications use the SetD1PID and SetD0PID fields of this register to program either DATA0 or DATA1 PID. 1'b0: DATA0 1'b1: DATA1 This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. 1'b0 RO Even/Odd (Micro)Frame (EO_FrNum) In non-Scatter/Gather DMA mode: Applies to isochronous IN and OUT endpoints only. Indicates the (micro)frame number in which the core transmits/ receives isochronous data for this endpoint. The application must program the even/odd (micro) frame number in which it intends to transmit/receive isochronous data for this endpoint using the SetEvnFr and SetOddFr fields in this register. 1'b0: Even (micro)frame 1'b1: Odd (micro)frame When Scatter/Gather DMA mode is enabled, this field is reserved. The frame number in which to send data is provided in the transmit descriptor structure. The frame in which data is received is updated in receive descriptor structure. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RO	0x0

Bit	Name	Description	Access	Reset
15	usbactep	USB Active Endpoint (USBActEP) Indicates whether this endpoint is active in the current configuration and interface. The core clears this bit for all endpoints (other than EP 0) after detecting a USB reset. After receiving the SetConfiguration and SetInterface commands, the applica-tion must program endpoint registers accordingly and set this bit. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
10:0	mps	Maximum Packet Size (MPS) The application must program this field with the maximum packet size for the current logical endpoint. This value is in bytes.	RW	0x0

doepint4

Device OUT Endpoint 4 Interrupt Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00B88
i_usbotg_1_devgrp	0xFFB40800	0xFFB40B88

Offset: 0x388

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
stuppktrcvd RW 0x0	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	outpkterr RW 0x0	Reserved	back2backsetup RW 0x0	stsphsercvd RW 0x0	outtknepdis RW 0x0	setup RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0

doepint4 Fields

Bit	Name	Description	Access	Reset
15	stuppktrcvd	Setup Packet Received Applicable for Control OUT Endpoints in only in the Buffer DMA Mode Set by the DWC_otg core, this bit indicates that this buffer holds 8 bytes of setup data. There is only one Setup packet per buffer. On receiving a Setup packet, the DWC_otg core closes the buffer and disables the corresponding endpoint. The applica-tion has to re-enable the endpoint to receive any OUT data for the Control Transfer and reprogram the buffer start address. Note: Because of the above behavior, the DWC_otg core can receive any number of back to back setup packets and one buffer for every setup packet is used. 1'b0: No Setup packet received 1'b1: Setup packet received Reset: 1’b0	RW	0x0

Bit	Name	Description	Access	Reset
14	nyetintrpt	NYET Interrupt (NYETIntrpt) The core generates this interrupt when a NYET response is transmitted for a non isochronous OUT endpoint. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
13	nakintrpt	NAK Interrupt (NAKInterrupt) The core generates this interrupt when a NAK is transmitted or received by the device. In case of isochronous IN endpoints the interrupt gets generated when a zero length packet is transmitted due to un-availability of data in the TXFifo. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
12	bbleerr	NAK Interrupt (BbleErr) The core generates this interrupt when babble is received for the endpoint. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
11	pktdrpsts	Packet Drop Status (PktDrpSts) This bit indicates to the application that an ISOC OUT packet has been dropped. This bit does not have an associated mask bit and does not generate an interrupt. Dependency: This bit is valid in non Scatter/Gather DMA mode when periodic transfer interrupt feature is selected. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
9	bnaintr	BNA (Buffer Not Available) Interrupt (BNAIntr) This bit is valid only when Scatter/Gather DMA mode is enabled. The core generates this interrupt when the descriptor accessed is not ready For the Core to process, such as Host busy or DMA done **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
8	outpkterr	OUT Packet Error (OutPktErr) Applies to OUT endpoints Only This interrupt is valid only when thresholding is enabled. This interrupt is asserted when the core detects an overflow or a CRC error For non-Isochronous OUT packet. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
6	back2backsetup	Back-to-Back SETUP Packets Received (Back2BackSETup) Applies to Control OUT endpoints only. This bit indicates that the core has received more than three back-to-back SETUP packets For this particular endpoint. For information about handling this interrupt, **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
5	stsphsercvd	Status Phase Received For Control Write (StsPhseRcvd) This interrupt is valid only For Control OUT endpoints and only in Scatter Gather DMA mode. This interrupt is generated only after the core has transferred all the data that the host has sent during the data phase of a control write transfer, to the system memory buffer. The interrupt indicates to the application that the host has switched from data phase to the status phase of a Control Write transfer. The application can use this interrupt to ACK or STALL the Status phase, after it has decoded the data phase. This is applicable only in Case of Scatter Gather DMA mode. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
4	outtknepdis	OUT Token Received When Endpoint Disabled (OUTTknEPdis) Applies only to control OUT endpoints. Indicates that an OUT token was received when the endpoint was not yet enabled. This interrupt is asserted on the endpoint For which the OUT token was received. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
3	setup	SETUP Phase Done (SetUp) Applies to control OUT endpoints only. Indicates that the SETUP phase For the control endpoint is complete and no more back-to-back SETUP packets were received For the current control transfer. On this interrupt, the application can decode the received SETUP data packet. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
2	ahberr	AHB Error (AHBErr) Applies to IN and OUT endpoints. This is generated only in Internal DMA mode when there is an AHB error during an AHB read/write. The application can read the corresponding endpoint DMA address register to get the error address. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
1	epdisbld	Endpoint Disabled Interrupt (EPDisbld) Applies to IN and OUT endpoints. This bit indicates that the endpoint is disabled per the application's request. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
0	xfercompl	Transfer Completed Interrupt (XferCompl) Applies to IN and OUT endpoints. When Scatter/Gather DMA mode is enabled - For IN endpoint this field indicates that the requested data from the descriptor is moved from external system memory to internal FIFO. - For OUT endpoint this field indicates that the requested data from the internal FIFO is moved to external system memory. This interrupt is generated only when the corresponding endpoint descriptor is closed, and the IOC bit For the corresponding descriptor is Set. When Scatter/Gather DMA mode is disabled, this field indicates that the programmed transfer is complete on the AHB as well as on the USB, For this endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

doeptsiz4

Device OUT Endpoint 4 Transfer Size Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00B90
i_usbotg_1_devgrp	0xFFB40800	0xFFB40B90

Offset: 0x390

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	rxdpid RO 0x0		pktcnt RW 0x0										xfersize RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
xfersize RW 0x0															

doeptsiz4 Fields

Bit	Name	Description	Access	Reset
30:29	rxdpid	Applies to isochronous OUT endpoints only. This is the data PID received in the last packet for this endpoint. 2'b00: DATA0 2'b01: DATA2 2'b10: DATA1 2'b11: MDATA SETUP Packet Count (SUPCnt) Applies to control OUT Endpoints only. This field specifies the number of back-to-back SETUP data packets the endpoint can receive. 2'b01: 1 packet 2'b10: 2 packets 2'b11: 3 packets **Value / Description** 0x0 DATA0 0x1 DATA2PACKET1 0x2 DATA1PACKET2 0x3 MDATAPACKET3	RO	0x0
28:19	pktcnt	Packet Count (PktCnt) This field is decremented to zero after a packet is written into the RxFIFO.	RW	0x0
18:0	xfersize	Transfer Size (XferSize) Indicates the transfer size in bytes For endpoint 0. The core interrupts the application only after it has exhausted the transfer size amount of data. The transfer size can be Set to the maximum packet size of the endpoint, to be interrupted at the end of each packet. The core decrements this field every time a packet is read from the RxFIFO and written to the external memory.	RW	0x0

doepdma4

Device OUT Endpoint 4 DMA Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00B94
i_usbotg_1_devgrp	0xFFB40800	0xFFB40B94

Offset: 0x394

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdma4 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
doepdma4 RW 0x0															

doepdma4 Fields

Bit	Name	Description	Access	Reset
31:0	doepdma4	Holds the start address of the external memory for storing or fetching endpoint data. Note: For control endpoints, this field stores control OUT data packets as well as SETUP transaction data packets. When more than three SETUP packets are received back-to-back, the SETUP data packet in the memory is overwritten. This register is incremented on every AHB transaction. The application can give only a DWORD-aligned address. When Scatter/Gather DMA mode is not enabled, the application programs the start address value in this field. When Scatter/Gather DMA mode is enabled, this field indicates the base pointer for the descriptor list.	RW	0x0

doepdmab4

Device OUT Endpoint 4 Buffer Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00B9C
i_usbotg_1_devgrp	0xFFB40800	0xFFB40B9C

Offset: 0x39C

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdmab4 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
doepdmab4 RO 0x0															

doepdmab4 Fields

Bit	Name	Description	Access	Reset
31:0	doepdmab4	Holds the current buffer address.This register is updated as and when the data transfer for the corresponding end point is in progress. This register is present only in Scatter/Gather DMA mode. Otherwise this field is reserved.	RO	0x0

doepctl5

Device Control OUT Endpoint 5 Control Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00BA0
i_usbotg_1_devgrp	0xFFB40800	0xFFB40BA0

Offset: 0x3A0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	Reserved				stall RW 0x0	snp RW 0x0	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
usbactep RW 0x0	Reserved				mps RW 0x0										

doepctl5 Fields

Bit	Name	Description	Access	Reset
31	epena	Endpoint Enable (EPEna) Applies to IN and OUT endpoints. When Scatter/Gather DMA mode is enabled, For IN endpoints this bit indicates that the descriptor structure and data buffer with data ready to transmit is setup. For OUT endpoint it indicates that the descriptor structure and data buffer to receive data is setup. When Scatter/Gather DMA mode is enabledsuch as for buffer-pointer based DMA mode: - For IN endpoints, this bit indicates that data is ready to be transmitted on the endpoint. - For OUT endpoints, this bit indicates that the application has allocated the memory to start receiving data from the USB. - The core clears this bit before setting any of the following interrupts on this endpoint: SETUP Phase Done Endpoint Disabled Transfer Completed Note: For control endpoints in DMA mode, this bit must be set to be able to transfer SETUP data packets in memory. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
30	epdis	Endpoint Disable (EPDis) Applies to IN and OUT endpoints. The application sets this bit to stop transmitting/receiving data on an endpoint, even before the transfer for that endpoint is complete. The application must wait for the Endpoint Disabled interrupt before treating the endpoint as disabled. The core clears this bit before setting the Endpoint Disabled interrupt. The application must set this bit only if Endpoint Enable is already set for this endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
29	setd1pid	Set DATA1 PID (SetD1PID) Applies to interrupt/bulk IN and OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA1. This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. Set Odd (micro)frame (SetOddFr) Applies to isochronous IN and OUT endpoints only. Writing to this field sets the Even/Odd (micro)frame (EO_FrNum) field to odd (micro)frame. This field is not applicable for Scatter/Gather DMA mode. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	WO	0x0

Bit	Name	Description	Access	Reset
28	setd0pid	Set DATA0 PID (SetD0PID) Applies to interrupt/bulk IN and OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA0. This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. 1'b0 WO In non-Scatter/Gather DMA mode: Set Even (micro)frame (SetEvenFr) Applies to isochronous IN and OUT endpoints only. Writing to this field sets the Even/Odd (micro)frame (EO_FrNum) field to even (micro) frame. When Scatter/Gather DMA mode is enabled, this field is reserved. The frame number in which to send data is in the transmit descriptor structure. The frame in which to receive data is updated in receive descriptor structure. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	WO	0x0
27	snak	Set NAK (SNAK) A write to this bit sets the NAK bit For the endpoint. Using this bit, the application can control the transmission of NAK handshakes on an endpoint. The core can also Set this bit For an endpoint after a SETUP packet is received on that endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	WO	0x0

Bit	Name	Description	Access	Reset
26	cnak	Clear NAK (CNAK) A write to this bit clears the NAK bit For the endpoint. Value: 0x0 — INACTIVE Value: 0x1 — ACTIVE	WO	0x0
21	stall	STALL Handshake (Stall) Applies to non-control, non-isochronous IN and OUT endpoints only. The application sets this bit to stall all tokens from the USB host to this endpoint. If a NAK bit, Global Non-periodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority. Only the application can clear this bit, never the core. 1'b0 R_W Applies to control endpoints only. The application can only set this bit, and the core clears it, when a SETUP token is received for this endpoint. If a NAK bit, Global Non-periodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority. Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake. Value: 0x0 — INACTIVE Value: 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
20	snp	Snoop Mode (Snp) Applies to OUT endpoints only. This bit configures the endpoint to Snoop mode. In Snoop mode, the core does not check the correctness of OUT packets before transferring them to application memory. **Value** **Description** 0x0 DISABLE 0x1 ENABLE	RW	0x0
19:18	eptype	Endpoint Type (EPType) This is the transfer type supported by this logical endpoint. 2'b00: Control 2'b01: Isochronous 2'b10: Bulk 2'b11: Interrupt **Value** **Description** 0x0 CONTROL 0x1 ISOCHRONOUS 0x2 BULK 0x3 INTERRUP	RW	0x0

Bit	Name	Description	Access	Reset
17	naksts	NAK Status (NAKSts) Indicates the following: 1'b0: The core is transmitting non-NAK handshakes based on the FIFO status. 1'b1: The core is transmitting NAK handshakes on this endpoint. When either the application or the core sets this bit: The core stops receiving any data on an OUT endpoint, even if there is space in the RxFIFO to accommodate the incoming packet. For non-isochronous IN endpoints: The core stops transmitting any data on an IN endpoint, even if there data is available in the TxFIFO. For isochronous IN endpoints: The core sends out a zero-length data packet, even if there data is available in the TxFIFO. Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake. **Value** — **Description** 0x0 — NONNAK 0x1 — NAK	RO	0x0

Bit	Name	Description	Access	Reset
16	dpid	Endpoint Data PID (DPID) Applies to interrupt/bulk IN and OUT endpoints only. Contains the PID of the packet to be received or transmitted on this endpoint. The application must program the PID of the first packet to be received or transmitted on this endpoint, after the endpoint is activated. The applications use the SetD1PID and SetD0PID fields of this register to program either DATA0 or DATA1 PID. 1'b0: DATA0 1'b1: DATA1 This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. 1'b0 RO Even/Odd (Micro)Frame (EO_FrNum) In non-Scatter/Gather DMA mode: Applies to isochronous IN and OUT endpoints only. Indicates the (micro)frame number in which the core transmits/ receives isochronous data for this endpoint. The application must program the even/odd (micro) frame number in which it intends to transmit/receive isochronous data for this endpoint using the SetEvnFr and SetOddFr fields in this register. 1'b0: Even (micro)frame 1'b1: Odd (micro)frame When Scatter/Gather DMA mode is enabled, this field is reserved. The frame number in which to send data is provided in the transmit descriptor structure. The frame in which data is received is updated in receive descriptor structure. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RO	0x0

Bit	Name	Description	Access	Reset
15	usbactep	USB Active Endpoint (USBActEP) Indicates whether this endpoint is active in the current configuration and interface. The core clears this bit for all endpoints (other than EP 0) after detecting a USB reset. After receiving the SetConfiguration and SetInterface commands, the applica-tion must program endpoint registers accordingly and set this bit. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
10:0	mps	Maximum Packet Size (MPS) The application must program this field with the maximum packet size for the current logical endpoint. This value is in bytes.	RW	0x0

doepint5

Device OUT Endpoint 5 Interrupt Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00BA8
i_usbotg_1_devgrp	0xFFB40800	0xFFB40BA8

Offset: 0x3A8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
stuppktrcvd RW 0x0	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	outpkterr RW 0x0	Reserved	back2backsetup RW 0x0	stsphsercvd RW 0x0	outtknepdis RW 0x0	setup RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0

doepint5 Fields

Bit	Name	Description	Access	Reset
15	stuppktrcvd	Setup Packet Received Applicable for Control OUT Endpoints in only in the Buffer DMA Mode Set by the DWC_otg core, this bit indicates that this buffer holds 8 bytes of setup data. There is only one Setup packet per buffer. On receiving a Setup packet, the DWC_otg core closes the buffer and disables the corresponding endpoint. The applica-tion has to re-enable the endpoint to receive any OUT data for the Control Transfer and reprogram the buffer start address. Note: Because of the above behavior, the DWC_otg core can receive any number of back to back setup packets and one buffer for every setup packet is used. 1'b0: No Setup packet received 1'b1: Setup packet received Reset: 1’b0	RW	0x0

Bit	Name	Description	Access	Reset
14	nyetintrpt	NYET Interrupt (NYETIntrpt) The core generates this interrupt when a NYET response is transmitted for a non isochronous OUT endpoint. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
13	nakintrpt	NAK Interrupt (NAKInterrupt) The core generates this interrupt when a NAK is transmitted or received by the device. In case of isochronous IN endpoints the interrupt gets generated when a zero length packet is transmitted due to un-availability of data in the TXFifo. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
12	bbleerr	NAK Interrupt (BbleErr) The core generates this interrupt when babble is received for the endpoint. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
11	pktdrpsts	Packet Drop Status (PktDrpSts) This bit indicates to the application that an ISOC OUT packet has been dropped. This bit does not have an associated mask bit and does not generate an interrupt. Dependency: This bit is valid in non Scatter/Gather DMA mode when periodic transfer interrupt feature is selected. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
9	bnaintr	BNA (Buffer Not Available) Interrupt (BNAIntr) This bit is valid only when Scatter/Gather DMA mode is enabled. The core generates this interrupt when the descriptor accessed is not ready For the Core to process, such as Host busy or DMA done **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
8	outpkterr	OUT Packet Error (OutPktErr) Applies to OUT endpoints Only This interrupt is valid only when thresholding is enabled. This interrupt is asserted when the core detects an overflow or a CRC error For non-Isochronous OUT packet. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
6	back2backsetup	Back-to-Back SETUP Packets Received (Back2BackSETup) Applies to Control OUT endpoints only. This bit indicates that the core has received more than three back-to-back SETUP packets For this particular endpoint. For information about handling this interrupt, **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
5	stsphsercvd	Status Phase Received For Control Write (StsPhseRcvd) This interrupt is valid only For Control OUT endpoints and only in Scatter Gather DMA mode. This interrupt is generated only after the core has transferred all the data that the host has sent during the data phase of a control write transfer, to the system memory buffer. The interrupt indicates to the application that the host has switched from data phase to the status phase of a Control Write transfer. The application can use this interrupt to ACK or STALL the Status phase, after it has decoded the data phase. This is applicable only in Case of Scatter Gather DMA mode. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
4	outtknepdis	OUT Token Received When Endpoint Disabled (OUTTknEPdis) Applies only to control OUT endpoints. Indicates that an OUT token was received when the endpoint was not yet enabled. This interrupt is asserted on the endpoint For which the OUT token was received. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
3	setup	SETUP Phase Done (SetUp) Applies to control OUT endpoints only. Indicates that the SETUP phase For the control endpoint is complete and no more back-to-back SETUP packets were received For the current control transfer. On this interrupt, the application can decode the received SETUP data packet. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
2	ahberr	AHB Error (AHBErr) Applies to IN and OUT endpoints. This is generated only in Internal DMA mode when there is an AHB error during an AHB read/write. The application can read the corresponding endpoint DMA address register to get the error address. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
1	epdisbld	Endpoint Disabled Interrupt (EPDisbld) Applies to IN and OUT endpoints. This bit indicates that the endpoint is disabled per the application's request. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
0	xfercompl	Transfer Completed Interrupt (XferCompl) Applies to IN and OUT endpoints. When Scatter/Gather DMA mode is enabled - For IN endpoint this field indicates that the requested data from the descriptor is moved from external system memory to internal FIFO. - For OUT endpoint this field indicates that the requested data from the internal FIFO is moved to external system memory. This interrupt is generated only when the corresponding endpoint descriptor is closed, and the IOC bit For the corresponding descriptor is Set. When Scatter/Gather DMA mode is disabled, this field indicates that the programmed transfer is complete on the AHB as well as on the USB, For this endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

doeptsiz5

Device OUT Endpoint 5 Transfer Size Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00BB0
i_usbotg_1_devgrp	0xFFB40800	0xFFB40BB0

Offset: 0x3B0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	rxdpid RO 0x0		pktcnt RW 0x0										xfersize RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
xfersize RW 0x0															

doeptsiz5 Fields

Bit	Name	Description	Access	Reset
30:29	rxdpid	Applies to isochronous OUT endpoints only. This is the data PID received in the last packet for this endpoint. 2'b00: DATA0 2'b01: DATA2 2'b10: DATA1 2'b11: MDATA SETUP Packet Count (SUPCnt) Applies to control OUT Endpoints only. This field specifies the number of back-to-back SETUP data packets the endpoint can receive. 2'b01: 1 packet 2'b10: 2 packets 2'b11: 3 packets **Value / Description** 0x0 DATA0 0x1 DATA2PACKET1 0x2 DATA1PACKET2 0x3 MDATAPACKET3	RO	0x0
28:19	pktcnt	Packet Count (PktCnt) This field is decremented to zero after a packet is written into the RxFIFO.	RW	0x0
18:0	xfersize	Transfer Size (XferSize) Indicates the transfer size in bytes For endpoint 0. The core interrupts the application only after it has exhausted the transfer size amount of data. The transfer size can be Set to the maximum packet size of the endpoint, to be interrupted at the end of each packet. The core decrements this field every time a packet is read from the RxFIFO and written to the external memory.	RW	0x0

doepdma5

Device OUT Endpoint 5 DMA Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00BB4
i_usbotg_1_devgrp	0xFFB40800	0xFFB40BB4

Offset: 0x3B4

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdma5 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
doepdma5 RW 0x0															

doepdma5 Fields

Bit	Name	Description	Access	Reset
31:0	doepdma5	Holds the start address of the external memory for storing or fetching endpoint data. Note: For control endpoints, this field stores control OUT data packets as well as SETUP transaction data packets. When more than three SETUP packets are received back-to-back, the SETUP data packet in the memory is overwritten. This register is incremented on every AHB transaction. The application can give only a DWORD-aligned address. When Scatter/Gather DMA mode is not enabled, the application programs the start address value in this field. When Scatter/Gather DMA mode is enabled, this field indicates the base pointer for the descriptor list.	RW	0x0

doepdmab5

Device OUT Endpoint 5 Buffer Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00BBC
i_usbotg_1_devgrp	0xFFB40800	0xFFB40BBC

Offset: 0x3BC

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdmab5 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
doepdmab5 RO 0x0															

doepdmab5 Fields

Bit	Name	Description	Access	Reset
31:0	doepdmab5	Holds the current buffer address.This register is updated as and when the data transfer for the corresponding end point is in progress. This register is present only in Scatter/Gather DMA mode. Otherwise this field is reserved.	RO	0x0

doepctl6

Device Control OUT Endpoint 6 Control Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00BC0
i_usbotg_1_devgrp	0xFFB40800	0xFFB40BC0

Offset: 0x3C0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	Reserved				stall RW 0x0	snp RW 0x0	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
usbactep RW 0x0	Reserved				mps RW 0x0										

doepctl6 Fields

Bit	Name	Description	Access	Reset
31	epena	Endpoint Enable (EPEna) Applies to IN and OUT endpoints. When Scatter/Gather DMA mode is enabled, For IN endpoints this bit indicates that the descriptor structure and data buffer with data ready to transmit is setup. For OUT endpoint it indicates that the descriptor structure and data buffer to receive data is setup. When Scatter/Gather DMA mode is enabledsuch as for buffer-pointer based DMA mode: - For IN endpoints, this bit indicates that data is ready to be transmitted on the endpoint. - For OUT endpoints, this bit indicates that the application has allocated the memory to start receiving data from the USB. - The core clears this bit before setting any of the following interrupts on this endpoint: SETUP Phase Done Endpoint Disabled Transfer Completed Note: For control endpoints in DMA mode, this bit must be set to be able to transfer SETUP data packets in memory. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
30	epdis	Endpoint Disable (EPDis) Applies to IN and OUT endpoints. The application sets this bit to stop transmitting/receiving data on an endpoint, even before the transfer for that endpoint is complete. The application must wait for the Endpoint Disabled interrupt before treating the endpoint as disabled. The core clears this bit before setting the Endpoint Disabled interrupt. The application must set this bit only if Endpoint Enable is already set for this endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
29	setd1pid	Set DATA1 PID (SetD1PID) Applies to interrupt/bulk IN and OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA1. This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. Set Odd (micro)frame (SetOddFr) Applies to isochronous IN and OUT endpoints only. Writing to this field sets the Even/Odd (micro)frame (EO_FrNum) field to odd (micro)frame. This field is not applicable for Scatter/Gather DMA mode. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	WO	0x0

Bit	Name	Description	Access	Reset
28	setd0pid	Set DATA0 PID (SetD0PID) Applies to interrupt/bulk IN and OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA0. This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. 1'b0 WO In non-Scatter/Gather DMA mode: Set Even (micro)frame (SetEvenFr) Applies to isochronous IN and OUT endpoints only. Writing to this field sets the Even/Odd (micro)frame (EO_FrNum) field to even (micro) frame. When Scatter/Gather DMA mode is enabled, this field is reserved. The frame number in which to send data is in the transmit descriptor structure. The frame in which to receive data is updated in receive descriptor structure. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	WO	0x0
27	snak	Set NAK (SNAK) A write to this bit sets the NAK bit For the endpoint. Using this bit, the application can control the transmission of NAK handshakes on an endpoint. The core can also Set this bit For an endpoint after a SETUP packet is received on that endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	WO	0x0

Bit	Name	Description	Access	Reset
26	cnak	Clear NAK (CNAK) A write to this bit clears the NAK bit For the endpoint. Value: 0x0 — INACTIVE Value: 0x1 — ACTIVE	WO	0x0
21	stall	STALL Handshake (Stall) Applies to non-control, non-isochronous IN and OUT endpoints only. The application sets this bit to stall all tokens from the USB host to this endpoint. If a NAK bit, Global Non-periodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority. Only the application can clear this bit, never the core. 1'b0 R_W Applies to control endpoints only. The application can only set this bit, and the core clears it, when a SETUP token is received for this endpoint. If a NAK bit, Global Non-periodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority. Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake. Value: 0x0 — INACTIVE Value: 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
20	snp	Snoop Mode (Snp) Applies to OUT endpoints only. This bit configures the endpoint to Snoop mode. In Snoop mode, the core does not check the correctness of OUT packets before transferring them to application memory. **Value** **Description** 0x0 DISABLE 0x1 ENABLE	RW	0x0
19:18	eptype	Endpoint Type (EPType) This is the transfer type supported by this logical endpoint. 2'b00: Control 2'b01: Isochronous 2'b10: Bulk 2'b11: Interrupt **Value** **Description** 0x0 CONTROL 0x1 ISOCHRONOUS 0x2 BULK 0x3 INTERRUP	RW	0x0

Bit	Name	Description	Access	Reset
17	naksts	NAK Status (NAKSts) Indicates the following: 1'b0: The core is transmitting non-NAK handshakes based on the FIFO status. 1'b1: The core is transmitting NAK handshakes on this endpoint. When either the application or the core sets this bit: The core stops receiving any data on an OUT endpoint, even if there is space in the RxFIFO to accommodate the incoming packet. For non-isochronous IN endpoints: The core stops transmitting any data on an IN endpoint, even if there data is available in the TxFIFO. For isochronous IN endpoints: The core sends out a zero-length data packet, even if there data is available in the TxFIFO. Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake. **Value** — **Description** 0x0 — NONNAK 0x1 — NAK	RO	0x0

Bit	Name	Description	Access	Reset
16	dpid	Endpoint Data PID (DPID) Applies to interrupt/bulk IN and OUT endpoints only. Contains the PID of the packet to be received or transmitted on this endpoint. The application must program the PID of the first packet to be received or transmitted on this endpoint, after the endpoint is activated. The applications use the SetD1PID and SetD0PID fields of this register to program either DATA0 or DATA1 PID. 1'b0: DATA0 1'b1: DATA1 This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. 1'b0 RO Even/Odd (Micro)Frame (EO_FrNum) In non-Scatter/Gather DMA mode: Applies to isochronous IN and OUT endpoints only. Indicates the (micro)frame number in which the core transmits/receives isochronous data for this endpoint. The application must program the even/odd (micro) frame number in which it intends to transmit/receive isochronous data for this endpoint using the SetEvnFr and SetOddFr fields in this register. 1'b0: Even (micro)frame 1'b1: Odd (micro)frame When Scatter/Gather DMA mode is enabled, this field is reserved. The frame number in which to send data is provided in the transmit descriptor structure. The frame in which data is received is updated in receive descriptor structure. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RO	0x0

Bit	Name	Description	Access	Reset
15	usbactep	USB Active Endpoint (USBActEP) Indicates whether this endpoint is active in the current configuration and interface. The core clears this bit for all endpoints (other than EP 0) after detecting a USB reset. After receiving the SetConfiguration and SetInterface commands, the applica-tion must program endpoint registers accordingly and set this bit. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
10:0	mps	Maximum Packet Size (MPS) The application must program this field with the maximum packet size for the current logical endpoint. This value is in bytes.	RW	0x0

doepint6

Device OUT Endpoint 6 Interrupt Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00BC8
i_usbotg_1_devgrp	0xFFB40800	0xFFB40BC8

Offset: 0x3C8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
stuppktrcvd RW 0x0	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	outpkterr RW 0x0	Reserved	back2backsetup RW 0x0	stsphsercvd RW 0x0	outtknepdis RW 0x0	setup RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0

doepint6 Fields

Bit	Name	Description	Access	Reset
15	stuppktrcvd	Setup Packet Received Applicable for Control OUT Endpoints in only in the Buffer DMA Mode Set by the DWC_otg core, this bit indicates that this buffer holds 8 bytes of setup data. There is only one Setup packet per buffer. On receiving a Setup packet, the DWC_otg core closes the buffer and disables the corresponding endpoint. The applica-tion has to re-enable the endpoint to receive any OUT data for the Control Transfer and reprogram the buffer start address. Note: Because of the above behavior, the DWC_otg core can receive any number of back to back setup packets and one buffer for every setup packet is used. 1'b0: No Setup packet received 1'b1: Setup packet received Reset: 1’b0	RW	0x0

Bit	Name	Description	Access	Reset
14	nyetintrpt	NYET Interrupt (NYETIntrpt) The core generates this interrupt when a NYET response is transmitted for a non isochronous OUT endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
13	nakintrpt	NAK Interrupt (NAKInterrupt) The core generates this interrupt when a NAK is transmitted or received by the device. In case of isochronous IN endpoints the interrupt gets generated when a zero length packet is transmitted due to un-availability of data in the TXFifo. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
12	bbleerr	NAK Interrupt (BbleErr) The core generates this interrupt when babble is received for the endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
11	pktdrpsts	Packet Drop Status (PktDrpSts) This bit indicates to the application that an ISOC OUT packet has been dropped. This bit does not have an associated mask bit and does not generate an interrupt. Dependency: This bit is valid in non Scatter/Gather DMA mode when periodic transfer interrupt feature is selected. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
9	bnaintr	BNA (Buffer Not Available) Interrupt (BNAIntr) This bit is valid only when Scatter/Gather DMA mode is enabled. The core generates this interrupt when the descriptor accessed is not ready For the Core to process, such as Host busy or DMA done **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
8	outpkterr	OUT Packet Error (OutPktErr) Applies to OUT endpoints Only This interrupt is valid only when thresholding is enabled. This interrupt is asserted when the core detects an overflow or a CRC error For non-Isochronous OUT packet. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
6	back2backsetup	Back-to-Back SETUP Packets Received (Back2BackSETup) Applies to Control OUT endpoints only. This bit indicates that the core has received more than three back-to-back SETUP packets For this particular endpoint. For information about handling this interrupt, **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
5	stsphsercvd	Status Phase Received For Control Write (StsPhseRcvd) This interrupt is valid only For Control OUT endpoints and only in Scatter Gather DMA mode. This interrupt is generated only after the core has transferred all the data that the host has sent during the data phase of a control write transfer, to the system memory buffer. The interrupt indicates to the application that the host has switched from data phase to the status phase of a Control Write transfer. The application can use this interrupt to ACK or STALL the Status phase, after it has decoded the data phase. This is applicable only in Case of Scatter Gather DMA mode. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
4	outtknepdis	OUT Token Received When Endpoint Disabled (OUTTknEPdis) Applies only to control OUT endpoints. Indicates that an OUT token was received when the endpoint was not yet enabled. This interrupt is asserted on the endpoint For which the OUT token was received. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
3	setup	SETUP Phase Done (SetUp) Applies to control OUT endpoints only. Indicates that the SETUP phase For the control endpoint is complete and no more back-to-back SETUP packets were received For the current control transfer. On this interrupt, the application can decode the received SETUP data packet. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
2	ahberr	AHB Error (AHBErr) Applies to IN and OUT endpoints. This is generated only in Internal DMA mode when there is an AHB error during an AHB read/write. The application can read the corresponding endpoint DMA address register to get the error address. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
1	epdisbld	Endpoint Disabled Interrupt (EPDisbld) Applies to IN and OUT endpoints. This bit indicates that the endpoint is disabled per the application's request. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
0	xfercompl	Transfer Completed Interrupt (XferCompl) Applies to IN and OUT endpoints. When Scatter/Gather DMA mode is enabled - For IN endpoint this field indicates that the requested data from the descriptor is moved from external system memory to internal FIFO. - For OUT endpoint this field indicates that the requested data from the internal FIFO is moved to external system memory. This interrupt is generated only when the corresponding endpoint descriptor is closed, and the IOC bit For the corresponding descriptor is Set. When Scatter/Gather DMA mode is disabled, this field indicates that the programmed transfer is complete on the AHB as well as on the USB, For this endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

doeptsiz6

Device OUT Endpoint 6 Transfer Size Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00BD0
i_usbotg_1_devgrp	0xFFB40800	0xFFB40BD0

Offset: 0x3D0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	rxdpid RO 0x0		pktcnt RW 0x0										xfersize RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
xfersize RW 0x0															

doeptsiz6 Fields

Bit	Name	Description	Access	Reset
30:29	rxdpid	Applies to isochronous OUT endpoints only. This is the data PID received in the last packet for this endpoint. 2'b00: DATA0 2'b01: DATA2 2'b10: DATA1 2'b11: MDATA SETUP Packet Count (SUPCnt) Applies to control OUT Endpoints only. This field specifies the number of back-to-back SETUP data packets the endpoint can receive. 2'b01: 1 packet 2'b10: 2 packets 2'b11: 3 packets **Value** **Description** 0x0 DATA0 0x1 DATA2PACKET1 0x2 DATA1PACKET2 0x3 MDATAPACKET3	RO	0x0
28:19	pktcnt	Packet Count (PktCnt) This field is decremented to zero after a packet is written into the RxFIFO.	RW	0x0
18:0	xfersize	Transfer Size (XferSize) Indicates the transfer size in bytes For endpoint 0. The core interrupts the application only after it has exhausted the transfer size amount of data. The transfer size can be Set to the maximum packet size of the endpoint, to be interrupted at the end of each packet. The core decrements this field every time a packet is read from the RxFIFO and written to the external memory.	RW	0x0

doepdma6

Device OUT Endpoint 6 DMA Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00BD4
i_usbotg_1_devgrp	0xFFB40800	0xFFB40BD4

Offset: 0x3D4

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdma6 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
doepdma6 RW 0x0															

doepdma6 Fields

Bit	Name	Description	Access	Reset
31:0	doepdma6	Holds the start address of the external memory for storing or fetching endpoint data. Note: For control endpoints, this field stores control OUT data packets as well as SETUP transaction data packets. When more than three SETUP packets are received back-to-back, the SETUP data packet in the memory is overwritten. This register is incremented on every AHB transaction. The application can give only a DWORD-aligned address. When Scatter/Gather DMA mode is not enabled, the application programs the start address value in this field. When Scatter/Gather DMA mode is enabled, this field indicates the base pointer for the descriptor list.	RW	0x0

doepdmab6

Device OUT Endpoint 6 Buffer Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00BDC
i_usbotg_1_devgrp	0xFFB40800	0xFFB40BDC

Offset: 0x3DC

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdmab6 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
doepdmab6 RO 0x0															

doepdmab6 Fields

Bit	Name	Description	Access	Reset
31:0	doepdmab6	Holds the current buffer address.This register is updated as and when the data transfer for the corresponding end point is in progress. This register is present only in Scatter/Gather DMA mode. Otherwise this field is reserved.	RO	0x0

doepctl7

Device Control OUT Endpoint 7 Control Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00BE0
i_usbotg_1_devgrp	0xFFB40800	0xFFB40BE0

Offset: 0x3E0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	Reserved				stall RW 0x0	snp RW 0x0	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
usbactep RW 0x0	Reserved				mps RW 0x0										

doepctl7 Fields

Bit	Name	Description	Access	Reset
31	epena	Endpoint Enable (EPEna) Applies to IN and OUT endpoints. When Scatter/Gather DMA mode is enabled, For IN endpoints this bit indicates that the descriptor structure and data buffer with data ready to transmit is setup. For OUT endpoint it indicates that the descriptor structure and data buffer to receive data is setup. When Scatter/Gather DMA mode is enabledsuch as for buffer-pointer based DMA mode: - For IN endpoints, this bit indicates that data is ready to be transmitted on the endpoint. - For OUT endpoints, this bit indicates that the application has allocated the memory to start receiving data from the USB. - The core clears this bit before setting any of the following interrupts on this endpoint: SETUP Phase Done Endpoint Disabled Transfer Completed Note: For control endpoints in DMA mode, this bit must be set to be able to transfer SETUP data packets in memory. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
30	epdis	Endpoint Disable (EPDis) Applies to IN and OUT endpoints. The application sets this bit to stop transmitting/receiving data on an endpoint, even before the transfer for that endpoint is complete. The application must wait for the Endpoint Disabled interrupt before treating the endpoint as disabled. The core clears this bit before setting the Endpoint Disabled interrupt. The application must set this bit only if Endpoint Enable is already set for this endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
29	setd1pid	Set DATA1 PID (SetD1PID) Applies to interrupt/bulk IN and OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA1. This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. Set Odd (micro)frame (SetOddFr) Applies to isochronous IN and OUT endpoints only. Writing to this field sets the Even/Odd (micro)frame (EO_FrNum) field to odd (micro)frame. This field is not applicable for Scatter/Gather DMA mode. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	WO	0x0

Bit	Name	Description	Access	Reset
28	setd0pid	Set DATA0 PID (SetD0PID) Applies to interrupt/bulk IN and OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA0. This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. 1'b0 WO In non-Scatter/Gather DMA mode: Set Even (micro)frame (SetEvenFr) Applies to isochronous IN and OUT endpoints only. Writing to this field sets the Even/Odd (micro)frame (EO_FrNum) field to even (micro) frame. When Scatter/Gather DMA mode is enabled, this field is reserved. The frame number in which to send data is in the transmit descriptor structure. The frame in which to receive data is updated in receive descriptor structure. **Value: Description** 0x0: DISABLED 0x1: ENABLED	WO	0x0
27	snak	Set NAK (SNAK) A write to this bit sets the NAK bit For the endpoint. Using this bit, the application can control the transmission of NAK handshakes on an endpoint. The core can also Set this bit For an endpoint after a SETUP packet is received on that endpoint. **Value: Description** 0x0: INACTIVE 0x1: ACTIVE	WO	0x0

Bit	Name	Description	Access	Reset
26	cnak	Clear NAK (CNAK) A write to this bit clears the NAK bit For the endpoint. Value: 0x0 — INACTIVE Value: 0x1 — ACTIVE	WO	0x0
21	stall	STALL Handshake (Stall) Applies to non-control, non-isochronous IN and OUT endpoints only. The application sets this bit to stall all tokens from the USB host to this endpoint. If a NAK bit, Global Non-periodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority. Only the application can clear this bit, never the core. 1'b0 R_W Applies to control endpoints only. The application can only set this bit, and the core clears it, when a SETUP token is received for this endpoint. If a NAK bit, Global Non-periodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority. Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake. Value: 0x0 — INACTIVE Value: 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
20	snp	Snoop Mode (Snp) Applies to OUT endpoints only. This bit configures the endpoint to Snoop mode. In Snoop mode, the core does not check the correctness of OUT packets before transferring them to application memory.	RW	0x0
19:18	eptype	Endpoint Type (EPType) This is the transfer type supported by this logical endpoint. 2'b00: Control 2'b01: Isochronous 2'b10: Bulk 2'b11: Interrupt **Value** — **Description** 0x0 — CONTROL 0x1 — ISOCHRONOUS 0x2 — BULK 0x3 — INTERRUP	RW	0x0

Bit	Name	Description	Access	Reset
17	naksts	NAK Status (NAKSts) Indicates the following: 1'b0: The core is transmitting non-NAK handshakes based on the FIFO status. 1'b1: The core is transmitting NAK handshakes on this endpoint. When either the application or the core sets this bit: The core stops receiving any data on an OUT endpoint, even if there is space in the RxFIFO to accommodate the incoming packet. For non-isochronous IN endpoints: The core stops transmitting any data on an IN endpoint, even if there data is available in the TxFIFO. For isochronous IN endpoints: The core sends out a zero-length data packet, even if there data is available in the TxFIFO. Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake. **Value** — **Description** 0x0 — NONNAK 0x1 — NAK	RO	0x0

Bit	Name	Description	Access	Reset
16	dpid	Endpoint Data PID (DPID) Applies to interrupt/bulk IN and OUT endpoints only. Contains the PID of the packet to be received or transmitted on this endpoint. The application must program the PID of the first packet to be received or transmitted on this endpoint, after the endpoint is activated. The applications use the SetD1PID and SetD0PID fields of this register to program either DATA0 or DATA1 PID. 1'b0: DATA0 1'b1: DATA1 This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. 1'b0 RO Even/Odd (Micro)Frame (EO_FrNum) In non-Scatter/Gather DMA mode: Applies to isochronous IN and OUT endpoints only. Indicates the (micro)frame number in which the core transmits/ receives isochronous data for this endpoint. The application must program the even/odd (micro) frame number in which it intends to transmit/receive isochronous data for this endpoint using the SetEvnFr and SetOddFr fields in this register. 1'b0: Even (micro)frame 1'b1: Odd (micro)frame When Scatter/Gather DMA mode is enabled, this field is reserved. The frame number in which to send data is provided in the transmit descriptor structure. The frame in which data is received is updated in receive descriptor structure. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RO	0x0

Bit	Name	Description	Access	Reset
15	usbactep	USB Active Endpoint (USBActEP) Indicates whether this endpoint is active in the current configuration and interface. The core clears this bit for all endpoints (other than EP 0) after detecting a USB reset. After receiving the SetConfiguration and SetInterface commands, the applica-tion must program endpoint registers accordingly and set this bit. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
10:0	mps	Maximum Packet Size (MPS) The application must program this field with the maximum packet size for the current logical endpoint. This value is in bytes.	RW	0x0

doepint7

Device OUT Endpoint 7 Interrupt Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00BE8
i_usbotg_1_devgrp	0xFFB40800	0xFFB40BE8

Offset: 0x3E8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
stuppktrcvd RW 0x0	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	outpkterr RW 0x0	Reserved	back2backsetup RW 0x0	stsphsercvd RW 0x0	outtknepdis RW 0x0	setup RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0

doepint7 Fields

Bit	Name	Description	Access	Reset
15	stuppktrcvd	Setup Packet Received Applicable for Control OUT Endpoints in only in the Buffer DMA Mode Set by the DWC_otg core, this bit indicates that this buffer holds 8 bytes of setup data. There is only one Setup packet per buffer. On receiving a Setup packet, the DWC_otg core closes the buffer and disables the corresponding endpoint. The applica-tion has to re-enable the endpoint to receive any OUT data for the Control Transfer and reprogram the buffer start address. Note: Because of the above behavior, the DWC_otg core can receive any number of back to back setup packets and one buffer for every setup packet is used. 1'b0: No Setup packet received 1'b1: Setup packet received Reset: 1’b0	RW	0x0

Bit	Name	Description	Access	Reset
14	nyetintrpt	NYET Interrupt (NYETIntrpt) The core generates this interrupt when a NYET response is transmitted for a non isochronous OUT endpoint. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
13	nakintrpt	NAK Interrupt (NAKInterrupt) The core generates this interrupt when a NAK is transmitted or received by the device. In case of isochronous IN endpoints the interrupt gets generated when a zero length packet is transmitted due to un-availability of data in the TXFifo. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
12	bbleerr	NAK Interrupt (BbleErr) The core generates this interrupt when babble is received for the endpoint. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
11	pktdrpsts	Packet Drop Status (PktDrpSts) This bit indicates to the application that an ISOC OUT packet has been dropped. This bit does not have an associated mask bit and does not generate an interrupt. Dependency: This bit is valid in non Scatter/Gather DMA mode when periodic transfer interrupt feature is selected. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
9	bnaintr	BNA (Buffer Not Available) Interrupt (BNAIntr) This bit is valid only when Scatter/Gather DMA mode is enabled. The core generates this interrupt when the descriptor accessed is not ready For the Core to process, such as Host busy or DMA done **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
8	outpkterr	OUT Packet Error (OutPktErr) Applies to OUT endpoints Only This interrupt is valid only when thresholding is enabled. This interrupt is asserted when the core detects an overflow or a CRC error For non-Isochronous OUT packet. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
6	back2backsetup	Back-to-Back SETUP Packets Received (Back2BackSETup) Applies to Control OUT endpoints only. This bit indicates that the core has received more than three back-to-back SETUP packets For this particular endpoint. For information about handling this interrupt, **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
5	stsphsercvd	Status Phase Received For Control Write (StsPhseRcvd) This interrupt is valid only For Control OUT endpoints and only in Scatter Gather DMA mode. This interrupt is generated only after the core has transferred all the data that the host has sent during the data phase of a control write transfer, to the system memory buffer. The interrupt indicates to the application that the host has switched from data phase to the status phase of a Control Write transfer. The application can use this interrupt to ACK or STALL the Status phase, after it has decoded the data phase. This is applicable only in Case of Scatter Gather DMA mode. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
4	outtknepdis	OUT Token Received When Endpoint Disabled (OUTTknEPdis) Applies only to control OUT endpoints. Indicates that an OUT token was received when the endpoint was not yet enabled. This interrupt is asserted on the endpoint For which the OUT token was received. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
3	setup	SETUP Phase Done (SetUp) Applies to control OUT endpoints only. Indicates that the SETUP phase For the control endpoint is complete and no more back-to-back SETUP packets were received For the current control transfer. On this interrupt, the application can decode the received SETUP data packet. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
2	ahberr	AHB Error (AHBErr) Applies to IN and OUT endpoints. This is generated only in Internal DMA mode when there is an AHB error during an AHB read/write. The application can read the corresponding endpoint DMA address register to get the error address. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
1	epdisbld	Endpoint Disabled Interrupt (EPDisbld) Applies to IN and OUT endpoints. This bit indicates that the endpoint is disabled per the application's request. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
0	xfercompl	Transfer Completed Interrupt (XferCompl) Applies to IN and OUT endpoints. When Scatter/Gather DMA mode is enabled - For IN endpoint this field indicates that the requested data from the descriptor is moved from external system memory to internal FIFO. - For OUT endpoint this field indicates that the requested data from the internal FIFO is moved to external system memory. This interrupt is generated only when the corresponding endpoint descriptor is closed, and the IOC bit For the corresponding descriptor is Set. When Scatter/Gather DMA mode is disabled, this field indicates that the programmed transfer is complete on the AHB as well as on the USB, For this endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

doeptsiz7

Device OUT Endpoint 7 Transfer Size Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00BF0
i_usbotg_1_devgrp	0xFFB40800	0xFFB40BF0

Offset: 0x3F0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	rxdpid RO 0x0		pktcnt RW 0x0										xfersize RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
xfersize RW 0x0															

doeptsiz7 Fields

Bit	Name	Description	Access	Reset
30:29	rxdpid	Applies to isochronous OUT endpoints only. This is the data PID received in the last packet for this endpoint. 2'b00: DATA0 2'b01: DATA2 2'b10: DATA1 2'b11: MDATA SETUP Packet Count (SUPCnt) Applies to control OUT Endpoints only. This field specifies the number of back-to-back SETUP data packets the endpoint can receive. 2'b01: 1 packet 2'b10: 2 packets 2'b11: 3 packets **Value** **Description** 0x0 DATA0 0x1 DATA2PACKET1 0x2 DATA1PACKET2 0x3 MDATAPACKET3	RO	0x0
28:19	pktcnt	Packet Count (PktCnt) This field is decremented to zero after a packet is written into the RxFIFO.	RW	0x0
18:0	xfersize	Transfer Size (XferSize) Indicates the transfer size in bytes For endpoint 0. The core interrupts the application only after it has exhausted the transfer size amount of data. The transfer size can be Set to the maximum packet size of the endpoint, to be interrupted at the end of each packet. The core decrements this field every time a packet is read from the RxFIFO and written to the external memory.	RW	0x0

doepdma7

Device OUT Endpoint 7 DMA Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00BF4
i_usbotg_1_devgrp	0xFFB40800	0xFFB40BF4

Offset: 0x3F4

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdma7 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
doepdma7 RW 0x0															

doepdma7 Fields

Bit	Name	Description	Access	Reset
31:0	doepdma7	Holds the start address of the external memory for storing or fetching endpoint data. Note: For control endpoints, this field stores control OUT data packets as well as SETUP transaction data packets. When more than three SETUP packets are received back-to-back, the SETUP data packet in the memory is overwritten. This register is incremented on every AHB transaction. The application can give only a DWORD-aligned address. When Scatter/Gather DMA mode is not enabled, the application programs the start address value in this field. When Scatter/Gather DMA mode is enabled, this field indicates the base pointer for the descriptor list.	RW	0x0

doepdmab7

Device OUT Endpoint 7 Buffer Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00BFC
i_usbotg_1_devgrp	0xFFB40800	0xFFB40BFC

Offset: 0x3FC

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdmab7 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
doepdmab7 RO 0x0															

doepdmab7 Fields

Bit	Name	Description	Access	Reset
31:0	doepdmab7	Holds the current buffer address.This register is updated as and when the data transfer for the corresponding end point is in progress. This register is present only in Scatter/Gather DMA mode. Otherwise this field is reserved.	RO	0x0

doepctl8

Device Control OUT Endpoint 8 Control Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00C00
i_usbotg_1_devgrp	0xFFB40800	0xFFB40C00

Offset: 0x400

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	Reserved				stall RW 0x0	snp RW 0x0	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
usbactep RW 0x0	Reserved				mps RW 0x0										

doepctl8 Fields

Bit	Name	Description	Access	Reset
31	epena	Endpoint Enable (EPEna) Applies to IN and OUT endpoints. When Scatter/Gather DMA mode is enabled, For IN endpoints this bit indicates that the descriptor structure and data buffer with data ready to transmit is setup. For OUT endpoint it indicates that the descriptor structure and data buffer to receive data is setup. When Scatter/Gather DMA mode is enabledsuch as for buffer-pointer based DMA mode: - For IN endpoints, this bit indicates that data is ready to be transmitted on the endpoint. - For OUT endpoints, this bit indicates that the application has allocated the memory to start receiving data from the USB. - The core clears this bit before setting any of the following interrupts on this endpoint: SETUP Phase Done Endpoint Disabled Transfer Completed Note: For control endpoints in DMA mode, this bit must be set to be able to transfer SETUP data packets in memory. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
30	epdis	Endpoint Disable (EPDis) Applies to IN and OUT endpoints. The application sets this bit to stop transmitting/receiving data on an endpoint, even before the transfer for that endpoint is complete. The application must wait for the Endpoint Disabled interrupt before treating the endpoint as disabled. The core clears this bit before setting the Endpoint Disabled interrupt. The application must set this bit only if Endpoint Enable is already set for this endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
29	setd1pid	Set DATA1 PID (SetD1PID) Applies to interrupt/bulk IN and OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA1. This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. Set Odd (micro)frame (SetOddFr) Applies to isochronous IN and OUT endpoints only. Writing to this field sets the Even/Odd (micro)frame (EO_FrNum) field to odd (micro)frame. This field is not applicable for Scatter/Gather DMA mode. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	WO	0x0

Bit	Name	Description	Access	Reset
28	setd0pid	Set DATA0 PID (SetD0PID) Applies to interrupt/bulk IN and OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA0. This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. 1'b0 WO In non-Scatter/Gather DMA mode: Set Even (micro)frame (SetEvenFr) Applies to isochronous IN and OUT endpoints only. Writing to this field sets the Even/Odd (micro)frame (EO_FrNum) field to even (micro) frame. When Scatter/Gather DMA mode is enabled, this field is reserved. The frame number in which to send data is in the transmit descriptor structure. The frame in which to receive data is updated in receive descriptor structure. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	WO	0x0
27	snak	Set NAK (SNAK) A write to this bit sets the NAK bit For the endpoint. Using this bit, the application can control the transmission of NAK handshakes on an endpoint. The core can also Set this bit For an endpoint after a SETUP packet is received on that endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	WO	0x0

Bit	Name	Description	Access	Reset
26	cnak	Clear NAK (CNAK) A write to this bit clears the NAK bit For the endpoint. Value: 0x0 — INACTIVE Value: 0x1 — ACTIVE	WO	0x0
21	stall	STALL Handshake (Stall) Applies to non-control, non-isochronous IN and OUT endpoints only. The application sets this bit to stall all tokens from the USB host to this endpoint. If a NAK bit, Global Non-periodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority. Only the application can clear this bit, never the core. 1'b0 R_W Applies to control endpoints only. The application can only set this bit, and the core clears it, when a SETUP token is received for this endpoint. If a NAK bit, Global Non-periodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority. Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake. Value: 0x0 — INACTIVE Value: 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
20	snp	Snoop Mode (Snp) Applies to OUT endpoints only. This bit configures the endpoint to Snoop mode. In Snoop mode, the core does not check the correctness of OUT packets before transferring them to application memory. **Value** **Description** 0x0 DISABLE 0x1 ENABLE	RW	0x0
19:18	eptype	Endpoint Type (EPType) This is the transfer type supported by this logical endpoint. 2'b00: Control 2'b01: Isochronous 2'b10: Bulk 2'b11: Interrupt **Value** **Description** 0x0 CONTROL 0x1 ISOCHRONOUS 0x2 BULK 0x3 INTERRUP	RW	0x0

Bit	Name	Description	Access	Reset
17	naksts	NAK Status (NAKSts) Indicates the following: 1'b0: The core is transmitting non-NAK handshakes based on the FIFO status. 1'b1: The core is transmitting NAK handshakes on this endpoint. When either the application or the core sets this bit: The core stops receiving any data on an OUT endpoint, even if there is space in the RxFIFO to accommodate the incoming packet. For non-isochronous IN endpoints: The core stops transmitting any data on an IN endpoint, even if there data is available in the TxFIFO. For isochronous IN endpoints: The core sends out a zero-length data packet, even if there data is available in the TxFIFO. Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake. **Value** — **Description** 0x0 — NONNAK 0x1 — NAK	RO	0x0

Bit	Name	Description	Access	Reset
16	dpid	Endpoint Data PID (DPID) Applies to interrupt/bulk IN and OUT endpoints only. Contains the PID of the packet to be received or transmitted on this endpoint. The application must program the PID of the first packet to be received or transmitted on this endpoint, after the endpoint is activated. The applications use the SetD1PID and SetD0PID fields of this register to program either DATA0 or DATA1 PID. 1'b0: DATA0 1'b1: DATA1 This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. 1'b0 RO Even/Odd (Micro)Frame (EO_FrNum) In non-Scatter/Gather DMA mode: Applies to isochronous IN and OUT endpoints only. Indicates the (micro)frame number in which the core transmits/ receives isochronous data for this endpoint. The application must program the even/odd (micro) frame number in which it intends to transmit/receive isochronous data for this endpoint using the SetEvnFr and SetOddFr fields in this register. 1'b0: Even (micro)frame 1'b1: Odd (micro)frame When Scatter/Gather DMA mode is enabled, this field is reserved. The frame number in which to send data is provided in the transmit descriptor structure. The frame in which data is received is updated in receive descriptor structure. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RO	0x0

Bit	Name	Description	Access	Reset
15	usbactep	USB Active Endpoint (USBActEP) Indicates whether this endpoint is active in the current configuration and interface. The core clears this bit for all endpoints (other than EP 0) after detecting a USB reset. After receiving the SetConfiguration and SetInterface commands, the applica-tion must program endpoint registers accordingly and set this bit. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
10:0	mps	Maximum Packet Size (MPS) The application must program this field with the maximum packet size for the current logical endpoint. This value is in bytes.	RW	0x0

doepint8

Device OUT Endpoint 8 Interrupt Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00C08
i_usbotg_1_devgrp	0xFFB40800	0xFFB40C08

Offset: 0x408

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
stuppktrcvd RW 0x0	nyetintrpt RW 0x0	nakintrpt RW 0x0	babblerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	outpkterr RW 0x0	Reserved	back2backsetup RW 0x0	stsphsercvd RW 0x0	outtknepdis RW 0x0	setup RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0

doepint8 Fields

Bit	Name	Description	Access	Reset
15	stuppktrcvd	Setup Packet Received Applicable for Control OUT Endpoints in only in the Buffer DMA Mode Set by the DWC_otg core, this bit indicates that this buffer holds 8 bytes of setup data. There is only one Setup packet per buffer. On receiving a Setup packet, the DWC_otg core closes the buffer and disables the corresponding endpoint. The application has to re-enable the endpoint to receive any OUT data for the Control Transfer and reprogram the buffer start address. Note: Because of the above behavior, the DWC_otg core can receive any number of back to back setup packets and one buffer for every setup packet is used. 1'b0: No Setup packet received 1'b1: Setup packet received Reset: 1’b0	RW	0x0

Bit	Name	Description	Access	Reset
14	nyetintrpt	NYET Interrupt (NYETIntrpt) The core generates this interrupt when a NYET response is transmitted for a non isochronous OUT endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
13	nakintrpt	NAK Interrupt (NAKInterrupt) The core generates this interrupt when a NAK is transmitted or received by the device. In case of isochronous IN endpoints the interrupt gets generated when a zero length packet is transmitted due to un-availability of data in the TXFifo. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
12	bbleerr	NAK Interrupt (BbleErr) The core generates this interrupt when babble is received for the endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
11	pktdrpsts	Packet Drop Status (PktDrpSts) This bit indicates to the application that an ISOC OUT packet has been dropped. This bit does not have an associated mask bit and does not generate an interrupt. Dependency: This bit is valid in non Scatter/Gather DMA mode when periodic transfer interrupt feature is selected. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
9	bnaintr	BNA (Buffer Not Available) Interrupt (BNAIntr) This bit is valid only when Scatter/Gather DMA mode is enabled. The core generates this interrupt when the descriptor accessed is not ready For the Core to process, such as Host busy or DMA done **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
8	outpkterr	OUT Packet Error (OutPktErr) Applies to OUT endpoints Only This interrupt is valid only when thresholding is enabled. This interrupt is asserted when the core detects an overflow or a CRC error For non-Isochronous OUT packet. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
6	back2backsetup	Back-to-Back SETUP Packets Received (Back2BackSETup) Applies to Control OUT endpoints only. This bit indicates that the core has received more than three back-to-back SETUP packets For this particular endpoint. For information about handling this interrupt, **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
5	stsphsercvd	Status Phase Received For Control Write (StsPhseRcvd) This interrupt is valid only For Control OUT endpoints and only in Scatter Gather DMA mode. This interrupt is generated only after the core has transferred all the data that the host has sent during the data phase of a control write transfer, to the system memory buffer. The interrupt indicates to the application that the host has switched from data phase to the status phase of a Control Write transfer. The application can use this interrupt to ACK or STALL the Status phase, after it has decoded the data phase. This is applicable only in Case of Scatter Gather DMA mode. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
4	outtknepdis	OUT Token Received When Endpoint Disabled (OUTTknEPdis) Applies only to control OUT endpoints. Indicates that an OUT token was received when the endpoint was not yet enabled. This interrupt is asserted on the endpoint For which the OUT token was received. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
3	setup	SETUP Phase Done (SetUp) Applies to control OUT endpoints only. Indicates that the SETUP phase For the control endpoint is complete and no more back-to-back SETUP packets were received For the current control transfer. On this interrupt, the application can decode the received SETUP data packet. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
2	ahberr	AHB Error (AHBErr) Applies to IN and OUT endpoints. This is generated only in Internal DMA mode when there is an AHB error during an AHB read/write. The application can read the corresponding endpoint DMA address register to get the error address. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
1	epdisbld	Endpoint Disabled Interrupt (EPDisbld) Applies to IN and OUT endpoints. This bit indicates that the endpoint is disabled per the application's request. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
0	xfercompl	Transfer Completed Interrupt (XferCompl) Applies to IN and OUT endpoints. When Scatter/Gather DMA mode is enabled - For IN endpoint this field indicates that the requested data from the descriptor is moved from external system memory to internal FIFO. - For OUT endpoint this field indicates that the requested data from the internal FIFO is moved to external system memory. This interrupt is generated only when the corresponding endpoint descriptor is closed, and the IOC bit For the corresponding descriptor is Set. When Scatter/Gather DMA mode is disabled, this field indicates that the programmed transfer is complete on the AHB as well as on the USB, For this endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

doeptsiz8

Device OUT Endpoint 8 Transfer Size Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00C10
i_usbotg_1_devgrp	0xFFB40800	0xFFB40C10

Offset: 0x410

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	rxdpid RO 0x0		pktcnt RW 0x0										xfersize RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
xfersize RW 0x0															

doeptsiz8 Fields

Bit	Name	Description	Access	Reset
30:29	rxdpid	Applies to isochronous OUT endpoints only. This is the data PID received in the last packet for this endpoint. 2'b00: DATA0 2'b01: DATA2 2'b10: DATA1 2'b11: MDATA SETUP Packet Count (SUPCnt) Applies to control OUT Endpoints only. This field specifies the number of back-to-back SETUP data packets the endpoint can receive. 2'b01: 1 packet 2'b10: 2 packets 2'b11: 3 packets **Value** **Description** 0x0 DATA0 0x1 DATA2PACKET1 0x2 DATA1PACKET2 0x3 MDATAPACKET3	RO	0x0
28:19	pktcnt	Packet Count (PktCnt) This field is decremented to zero after a packet is written into the RxFIFO.	RW	0x0
18:0	xfersize	Transfer Size (XferSize) Indicates the transfer size in bytes For endpoint 0. The core interrupts the application only after it has exhausted the transfer size amount of data. The transfer size can be Set to the maximum packet size of the endpoint, to be interrupted at the end of each packet. The core decrements this field every time a packet is read from the RxFIFO and written to the external memory.	RW	0x0

doepdma8

Device OUT Endpoint 8 DMA Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00C14
i_usbotg_1_devgrp	0xFFB40800	0xFFB40C14

Offset: 0x414

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdma8 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
doepdma8 RW 0x0															

doepdma8 Fields

Bit	Name	Description	Access	Reset
31:0	doepdma8	Holds the start address of the external memory for storing or fetching endpoint data. Note: For control endpoints, this field stores control OUT data packets as well as SETUP transaction data packets. When more than three SETUP packets are received back-to-back, the SETUP data packet in the memory is overwritten. This register is incremented on every AHB transaction. The application can give only a DWORD-aligned address. When Scatter/Gather DMA mode is not enabled, the application programs the start address value in this field. When Scatter/Gather DMA mode is enabled, this field indicates the base pointer for the descriptor list.	RW	0x0

doepdmab8

Device OUT Endpoint 8 Buffer Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00C1C
i_usbotg_1_devgrp	0xFFB40800	0xFFB40C1C

Offset: 0x41C

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdmab8 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
doepdmab8 RO 0x0															

doepdmab8 Fields

Bit	Name	Description	Access	Reset
31:0	doepdmab8	Holds the current buffer address.This register is updated as and when the data transfer for the corresponding end point is in progress. This register is present only in Scatter/Gather DMA mode. Otherwise this field is reserved.	RO	0x0

doepctl9

Device Control OUT Endpoint 9 Control Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00C20
i_usbotg_1_devgrp	0xFFB40800	0xFFB40C20

Offset: 0x420

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	Reserved				stall RW 0x0	snp RW 0x0	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
usbactep RW 0x0	Reserved				mps RW 0x0										

doepctl9 Fields

Bit	Name	Description	Access	Reset
31	epena	Endpoint Enable (EPEna) Applies to IN and OUT endpoints. When Scatter/Gather DMA mode is enabled, For IN endpoints this bit indicates that the descriptor structure and data buffer with data ready to transmit is setup. For OUT endpoint it indicates that the descriptor structure and data buffer to receive data is setup. When Scatter/Gather DMA mode is enabledsuch as for buffer-pointer based DMA mode: - For IN endpoints, this bit indicates that data is ready to be transmitted on the endpoint. - For OUT endpoints, this bit indicates that the application has allocated the memory to start receiving data from the USB. - The core clears this bit before setting any of the following interrupts on this endpoint: SETUP Phase Done Endpoint Disabled Transfer Completed Note: For control endpoints in DMA mode, this bit must be set to be able to transfer SETUP data packets in memory. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
30	epdis	Endpoint Disable (EPDis) Applies to IN and OUT endpoints. The application sets this bit to stop transmitting/receiving data on an endpoint, even before the transfer for that endpoint is complete. The application must wait for the Endpoint Disabled interrupt before treating the endpoint as disabled. The core clears this bit before setting the Endpoint Disabled interrupt. The application must set this bit only if Endpoint Enable is already set for this endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
29	setd1pid	Set DATA1 PID (SetD1PID) Applies to interrupt/bulk IN and OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA1. This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. Set Odd (micro)frame (SetOddFr) Applies to isochronous IN and OUT endpoints only. Writing to this field sets the Even/Odd (micro)frame (EO_FrNum) field to odd (micro)frame. This field is not applicable for Scatter/Gather DMA mode. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	WO	0x0

Bit	Name	Description	Access	Reset
28	setd0pid	Set DATA0 PID (SetD0PID) Applies to interrupt/bulk IN and OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA0. This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. 1'b0 WO In non-Scatter/Gather DMA mode: Set Even (micro)frame (SetEvenFr) Applies to isochronous IN and OUT endpoints only. Writing to this field sets the Even/Odd (micro)frame (EO_FrNum) field to even (micro) frame. When Scatter/Gather DMA mode is enabled, this field is reserved. The frame number in which to send data is in the transmit descriptor structure. The frame in which to receive data is updated in receive descriptor structure. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	WO	0x0
27	snak	Set NAK (SNAK) A write to this bit sets the NAK bit For the endpoint. Using this bit, the application can control the transmission of NAK handshakes on an endpoint. The core can also Set this bit For an endpoint after a SETUP packet is received on that endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	WO	0x0

Bit	Name	Description	Access	Reset
26	cnak	Clear NAK (CNAK) A write to this bit clears the NAK bit For the endpoint. Value: 0x0 — INACTIVE Value: 0x1 — ACTIVE	WO	0x0
21	stall	STALL Handshake (Stall) Applies to non-control, non-isochronous IN and OUT endpoints only. The application sets this bit to stall all tokens from the USB host to this endpoint. If a NAK bit, Global Non-periodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority. Only the application can clear this bit, never the core. 1'b0 R_W Applies to control endpoints only. The application can only set this bit, and the core clears it, when a SETUP token is received for this endpoint. If a NAK bit, Global Non-periodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority. Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake. Value: 0x0 — INACTIVE Value: 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
20	snp	Snoop Mode (Snp) Applies to OUT endpoints only. This bit configures the endpoint to Snoop mode. In Snoop mode, the core does not check the correctness of OUT packets before transferring them to application memory. **Value** — **Description** 0x0 — DISABLE 0x1 — ENABLE	RW	0x0
19:18	eptype	Endpoint Type (EPType) This is the transfer type supported by this logical endpoint. 2'b00: Control 2'b01: Isochronous 2'b10: Bulk 2'b11: Interrupt **Value** — **Description** 0x0 — CONTROL 0x1 — ISOCHRONOUS 0x2 — BULK 0x3 — INTERRUP	RW	0x0

Bit	Name	Description	Access	Reset
17	naksts	NAK Status (NAKSts) Indicates the following: 1'b0: The core is transmitting non-NAK handshakes based on the FIFO status. 1'b1: The core is transmitting NAK handshakes on this endpoint. When either the application or the core sets this bit: The core stops receiving any data on an OUT endpoint, even if there is space in the RxFIFO to accommodate the incoming packet. For non-isochronous IN endpoints: The core stops transmitting any data on an IN endpoint, even if there data is available in the TxFIFO. For isochronous IN endpoints: The core sends out a zero-length data packet, even if there data is available in the TxFIFO. Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake. **Value** — **Description** 0x0 — NONNAK 0x1 — NAK	RO	0x0

Bit	Name	Description	Access	Reset
16	dpid	Endpoint Data PID (DPID) Applies to interrupt/bulk IN and OUT endpoints only. Contains the PID of the packet to be received or transmitted on this endpoint. The application must program the PID of the first packet to be received or transmitted on this endpoint, after the endpoint is activated. The applications use the SetD1PID and SetD0PID fields of this register to program either DATA0 or DATA1 PID. 1'b0: DATA0 1'b1: DATA1 This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. 1'b0 RO Even/Odd (Micro)Frame (EO_FrNum) In non-Scatter/Gather DMA mode: Applies to isochronous IN and OUT endpoints only. Indicates the (micro)frame number in which the core transmits/ receives isochronous data for this endpoint. The application must program the even/odd (micro) frame number in which it intends to transmit/receive isochronous data for this endpoint using the SetEvnFr and SetOddFr fields in this register. 1'b0: Even (micro)frame 1'b1: Odd (micro)frame When Scatter/Gather DMA mode is enabled, this field is reserved. The frame number in which to send data is provided in the transmit descriptor structure. The frame in which data is received is updated in receive descriptor structure. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RO	0x0

Bit	Name	Description	Access	Reset
15	usbactep	USB Active Endpoint (USBActEP) Indicates whether this endpoint is active in the current configuration and interface. The core clears this bit for all endpoints (other than EP 0) after detecting a USB reset. After receiving the SetConfiguration and SetInterface commands, the applica-tion must program endpoint registers accordingly and set this bit. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
10:0	mps	Maximum Packet Size (MPS) The application must program this field with the maximum packet size for the current logical endpoint. This value is in bytes.	RW	0x0

doepint9

Device OUT Endpoint 9 Interrupt Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00C28
i_usbotg_1_devgrp	0xFFB40800	0xFFB40C28

Offset: 0x428

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
stuppktrcvd RW 0x0	nyetintrpt RW 0x0	nakintrpt RW 0x0	babblerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	outpkterr RW 0x0	Reserved	back2backsetup RW 0x0	stsphsercvd RW 0x0	outtknepdis RW 0x0	setup RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0

doepint9 Fields

Bit	Name	Description	Access	Reset
15	stuppktrcvd	Setup Packet Received Applicable for Control OUT Endpoints in only in the Buffer DMA Mode Set by the DWC_otg core, this bit indicates that this buffer holds 8 bytes of setup data. There is only one Setup packet per buffer. On receiving a Setup packet, the DWC_otg core closes the buffer and disables the corresponding endpoint. The application has to re-enable the endpoint to receive any OUT data for the Control Transfer and reprogram the buffer start address. Note: Because of the above behavior, the DWC_otg core can receive any number of back to back setup packets and one buffer for every setup packet is used. 1'b0: No Setup packet received 1'b1: Setup packet received Reset: 1’b0	RW	0x0

Bit	Name	Description	Access	Reset
14	nyetintrpt	NYET Interrupt (NYETIntrpt) The core generates this interrupt when a NYET response is transmitted for a non isochronous OUT endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
13	nakintrpt	NAK Interrupt (NAKInterrupt) The core generates this interrupt when a NAK is transmitted or received by the device. In case of isochronous IN endpoints the interrupt gets generated when a zero length packet is transmitted due to un-availability of data in the TXFifo. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
12	bbleerr	NAK Interrupt (BbleErr) The core generates this interrupt when babble is received for the endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
11	pktdrpsts	Packet Drop Status (PktDrpSts) This bit indicates to the application that an ISOC OUT packet has been dropped. This bit does not have an associated mask bit and does not generate an interrupt. Dependency: This bit is valid in non Scatter/Gather DMA mode when periodic transfer interrupt feature is selected. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
9	bnaintr	BNA (Buffer Not Available) Interrupt (BNAIntr) This bit is valid only when Scatter/Gather DMA mode is enabled. The core generates this interrupt when the descriptor accessed is not ready For the Core to process, such as Host busy or DMA done **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
8	outpkterr	OUT Packet Error (OutPktErr) Applies to OUT endpoints Only This interrupt is valid only when thresholding is enabled. This interrupt is asserted when the core detects an overflow or a CRC error For non-Isochronous OUT packet. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
6	back2backsetup	Back-to-Back SETUP Packets Received (Back2BackSETup) Applies to Control OUT endpoints only. This bit indicates that the core has received more than three back-to-back SETUP packets For this particular endpoint. For information about handling this interrupt, **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
5	stsphsercvd	Status Phase Received For Control Write (StsPhseRcvd) This interrupt is valid only For Control OUT endpoints and only in Scatter Gather DMA mode. This interrupt is generated only after the core has transferred all the data that the host has sent during the data phase of a control write transfer, to the system memory buffer. The interrupt indicates to the application that the host has switched from data phase to the status phase of a Control Write transfer. The application can use this interrupt to ACK or STALL the Status phase, after it has decoded the data phase. This is applicable only in Case of Scatter Gather DMA mode. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
4	outtknepdis	OUT Token Received When Endpoint Disabled (OUTTknEPdis) Applies only to control OUT endpoints. Indicates that an OUT token was received when the endpoint was not yet enabled. This interrupt is asserted on the endpoint For which the OUT token was received. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
3	setup	SETUP Phase Done (SetUp) Applies to control OUT endpoints only. Indicates that the SETUP phase For the control endpoint is complete and no more back-to-back SETUP packets were received For the current control transfer. On this interrupt, the application can decode the received SETUP data packet. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
2	ahberr	AHB Error (AHBErr) Applies to IN and OUT endpoints. This is generated only in Internal DMA mode when there is an AHB error during an AHB read/write. The application can read the corresponding endpoint DMA address register to get the error address. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
1	epdisbld	Endpoint Disabled Interrupt (EPDisbld) Applies to IN and OUT endpoints. This bit indicates that the endpoint is disabled per the application's request. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
0	xfercompl	Transfer Completed Interrupt (XferCompl) Applies to IN and OUT endpoints. When Scatter/Gather DMA mode is enabled - For IN endpoint this field indicates that the requested data from the descriptor is moved from external system memory to internal FIFO. - For OUT endpoint this field indicates that the requested data from the internal FIFO is moved to external system memory. This interrupt is generated only when the corresponding endpoint descriptor is closed, and the IOC bit For the corresponding descriptor is Set. When Scatter/Gather DMA mode is disabled, this field indicates that the programmed transfer is complete on the AHB as well as on the USB, For this endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

doeptsiz9

Device OUT Endpoint 9 Transfer Size Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00C30
i_usbotg_1_devgrp	0xFFB40800	0xFFB40C30

Offset: 0x430

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	rxdpid RO 0x0		pktcnt RW 0x0										xfersize RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
xfersize RW 0x0															

doeptsiz9 Fields

Bit	Name	Description	Access	Reset
30:29	rxdpid	Applies to isochronous OUT endpoints only. This is the data PID received in the last packet for this endpoint. 2'b00: DATA0 2'b01: DATA2 2'b10: DATA1 2'b11: MDATA SETUP Packet Count (SUPCnt) Applies to control OUT Endpoints only. This field specifies the number of back-to-back SETUP data packets the endpoint can receive. 2'b01: 1 packet 2'b10: 2 packets 2'b11: 3 packets **Value** **Description** 0x0 DATA0 0x1 DATA2PACKET1 0x2 DATA1PACKET2 0x3 MDATAPACKET3	RO	0x0
28:19	pktcnt	Packet Count (PktCnt) This field is decremented to zero after a packet is written into the RxFIFO.	RW	0x0
18:0	xfersize	Transfer Size (XferSize) Indicates the transfer size in bytes For endpoint 0. The core interrupts the application only after it has exhausted the transfer size amount of data. The transfer size can be Set to the maximum packet size of the endpoint, to be interrupted at the end of each packet. The core decrements this field every time a packet is read from the RxFIFO and written to the external memory.	RW	0x0

doepdma9

Device OUT Endpoint 9 DMA Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00C34
i_usbotg_1_devgrp	0xFFB40800	0xFFB40C34

Offset: 0x434

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdma9 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
doepdma9 RW 0x0															

doepdma9 Fields

Bit	Name	Description	Access	Reset
31:0	doepdma9	Holds the start address of the external memory for storing or fetching endpoint data. Note: For control endpoints, this field stores control OUT data packets as well as SETUP transaction data packets. When more than three SETUP packets are received back-to-back, the SETUP data packet in the memory is overwritten. This register is incremented on every AHB transaction. The application can give only a DWORD-aligned address. When Scatter/Gather DMA mode is not enabled, the application programs the start address value in this field. When Scatter/Gather DMA mode is enabled, this field indicates the base pointer for the descriptor list.	RW	0x0

doepdmab9

Device OUT Endpoint 9 Buffer Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00C3C
i_usbotg_1_devgrp	0xFFB40800	0xFFB40C3C

Offset: 0x43C

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdmab9 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
doepdmab9 RO 0x0															

doepdmab9 Fields

Bit	Name	Description	Access	Reset
31:0	doepdmab9	Holds the current buffer address.This register is updated as and when the data transfer for the corresponding end point is in progress. This register is present only in Scatter/Gather DMA mode. Otherwise this field is reserved.	RO	0x0

doepctl10

Device Control OUT Endpoint 10 Control Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00C40
i_usbotg_1_devgrp	0xFFB40800	0xFFB40C40

Offset: 0x440

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	Reserved				stall RW 0x0	snp RW 0x0	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
usbactep RW 0x0	Reserved				mps RW 0x0										

doepctl10 Fields

Bit	Name	Description	Access	Reset
31	epena	Endpoint Enable (EPEna) Applies to IN and OUT endpoints. When Scatter/Gather DMA mode is enabled, For IN endpoints this bit indicates that the descriptor structure and data buffer with data ready to transmit is setup. For OUT endpoint it indicates that the descriptor structure and data buffer to receive data is setup. When Scatter/Gather DMA mode is enabledsuch as for buffer-pointer based DMA mode: - For IN endpoints, this bit indicates that data is ready to be transmitted on the endpoint. - For OUT endpoints, this bit indicates that the application has allocated the memory to start receiving data from the USB. - The core clears this bit before setting any of the following interrupts on this endpoint: SETUP Phase Done Endpoint Disabled Transfer Completed Note: For control endpoints in DMA mode, this bit must be set to be able to transfer SETUP data packets in memory. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
30	epdis	Endpoint Disable (EPDis) Applies to IN and OUT endpoints. The application sets this bit to stop transmitting/receiving data on an endpoint, even before the transfer for that endpoint is complete. The application must wait for the Endpoint Disabled interrupt before treating the endpoint as disabled. The core clears this bit before setting the Endpoint Disabled interrupt. The application must set this bit only if Endpoint Enable is already set for this endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
29	setd1pid	Set DATA1 PID (SetD1PID) Applies to interrupt/bulk IN and OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA1. This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. Set Odd (micro)frame (SetOddFr) Applies to isochronous IN and OUT endpoints only. Writing to this field sets the Even/Odd (micro)frame (EO_FrNum) field to odd (micro)frame. This field is not applicable for Scatter/Gather DMA mode. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	WO	0x0

Bit	Name	Description	Access	Reset
28	setd0pid	Set DATA0 PID (SetD0PID) Applies to interrupt/bulk IN and OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA0. This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. 1'b0 WO In non-Scatter/Gather DMA mode: Set Even (micro)frame (SetEvenFr) Applies to isochronous IN and OUT endpoints only. Writing to this field sets the Even/Odd (micro)frame (EO_FrNum) field to even (micro) frame. When Scatter/Gather DMA mode is enabled, this field is reserved. The frame number in which to send data is in the transmit descriptor structure. The frame in which to receive data is updated in receive descriptor structure. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	WO	0x0
27	snak	Set NAK (SNAK) A write to this bit sets the NAK bit For the endpoint. Using this bit, the application can control the transmission of NAK handshakes on an endpoint. The core can also Set this bit For an endpoint after a SETUP packet is received on that endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	WO	0x0

Bit	Name	Description	Access	Reset
26	cnak	Clear NAK (CNAK) A write to this bit clears the NAK bit For the endpoint. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	WO	0x0
21	stall	STALL Handshake (Stall) Applies to non-control, non-isochronous IN and OUT endpoints only. The application sets this bit to stall all tokens from the USB host to this endpoint. If a NAK bit, Global Non-periodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority. Only the application can clear this bit, never the core. 1'b0 R_W Applies to control endpoints only. The application can only set this bit, and the core clears it, when a SETUP token is received for this endpoint. If a NAK bit, Global Non-periodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority. Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
20	snp	Snoop Mode (Snp) Applies to OUT endpoints only. This bit configures the endpoint to Snoop mode. In Snoop mode, the core does not check the correctness of OUT packets before transferring them to application memory. **Value** — **Description** 0x0 — DISABLE 0x1 — ENABLE	RW	0x0
19:18	eptype	Endpoint Type (EPType) This is the transfer type supported by this logical endpoint. 2'b00: Control 2'b01: Isochronous 2'b10: Bulk 2'b11: Interrupt **Value** — **Description** 0x0 — CONTROL 0x1 — ISOCHRONOUS 0x2 — BULK 0x3 — INTERRUP	RW	0x0

Bit	Name	Description	Access	Reset
17	naksts	NAK Status (NAKSts) Indicates the following: 1'b0: The core is transmitting non-NAK handshakes based on the FIFO status. 1'b1: The core is transmitting NAK handshakes on this endpoint. When either the application or the core sets this bit: The core stops receiving any data on an OUT endpoint, even if there is space in the RxFIFO to accommodate the incoming packet. For non-isochronous IN endpoints: The core stops transmitting any data on an IN endpoint, even if there data is available in the TxFIFO. For isochronous IN endpoints: The core sends out a zero-length data packet, even if there data is available in the TxFIFO. Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake. **Value** — **Description** 0x0 — NONNAK 0x1 — NAK	RO	0x0

Bit	Name	Description	Access	Reset
16	dpid	Endpoint Data PID (DPID) Applies to interrupt/bulk IN and OUT endpoints only. Contains the PID of the packet to be received or transmitted on this endpoint. The application must program the PID of the first packet to be received or transmitted on this endpoint, after the endpoint is activated. The applications use the SetD1PID and SetD0PID fields of this register to program either DATA0 or DATA1 PID. 1'b0: DATA0 1'b1: DATA1 This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. 1'b0 RO Even/Odd (Micro)Frame (EO_FrNum) In non-Scatter/Gather DMA mode: Applies to isochronous IN and OUT endpoints only. Indicates the (micro)frame number in which the core transmits/receives isochronous data for this endpoint. The application must program the even/odd (micro) frame number in which it intends to transmit/receive isochronous data for this endpoint using the SetEvnFr and SetOddFr fields in this register. 1'b0: Even (micro)frame 1'b1: Odd (micro)frame When Scatter/Gather DMA mode is enabled, this field is reserved. The frame number in which to send data is provided in the transmit descriptor structure. The frame in which data is received is updated in receive descriptor structure. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RO	0x0

Bit	Name	Description	Access	Reset
15	usbactep	USB Active Endpoint (USBActEP) Indicates whether this endpoint is active in the current configuration and interface. The core clears this bit for all endpoints (other than EP 0) after detecting a USB reset. After receiving the SetConfiguration and SetInterface commands, the applica-tion must program endpoint registers accordingly and set this bit. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
10:0	mps	Maximum Packet Size (MPS) The application must program this field with the maximum packet size for the current logical endpoint. This value is in bytes.	RW	0x0

doepint10

Device OUT Endpoint 10 Interrupt Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00C48
i_usbotg_1_devgrp	0xFFB40800	0xFFB40C48

Offset: 0x448

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
stuppktrcvd RW 0x0	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	outpkterr RW 0x0	Reserved	back2backsetup RW 0x0	stsphsercvd RW 0x0	outtknepdis RW 0x0	setup RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0

doepint10 Fields

Bit	Name	Description	Access	Reset
15	stuppktrcvd	Setup Packet Received Applicable for Control OUT Endpoints in only in the Buffer DMA Mode Set by the DWC_otg core, this bit indicates that this buffer holds 8 bytes of setup data. There is only one Setup packet per buffer. On receiving a Setup packet, the DWC_otg core closes the buffer and disables the corresponding endpoint. The applica-tion has to re-enable the endpoint to receive any OUT data for the Control Transfer and reprogram the buffer start address. Note: Because of the above behavior, the DWC_otg core can receive any number of back to back setup packets and one buffer for every setup packet is used. 1'b0: No Setup packet received 1'b1: Setup packet received Reset: 1’b0	RW	0x0

Bit	Name	Description	Access	Reset
14	nyetintrpt	NYET Interrupt (NYETIntrpt) The core generates this interrupt when a NYET response is transmitted for a non isochronous OUT endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
13	nakintrpt	NAK Interrupt (NAKInterrupt) The core generates this interrupt when a NAK is transmitted or received by the device. In case of isochronous IN endpoints the interrupt gets generated when a zero length packet is transmitted due to un-availability of data in the TXFifo. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
12	bbleerr	NAK Interrupt (BbleErr) The core generates this interrupt when babble is received for the endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
11	pktdrpsts	Packet Drop Status (PktDrpSts) This bit indicates to the application that an ISOC OUT packet has been dropped. This bit does not have an associated mask bit and does not generate an interrupt. Dependency: This bit is valid in non Scatter/Gather DMA mode when periodic transfer interrupt feature is selected. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
9	bnaintr	BNA (Buffer Not Available) Interrupt (BNAIntr) This bit is valid only when Scatter/Gather DMA mode is enabled. The core generates this interrupt when the descriptor accessed is not ready For the Core to process, such as Host busy or DMA done **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
8	outpkterr	OUT Packet Error (OutPktErr) Applies to OUT endpoints Only This interrupt is valid only when thresholding is enabled. This interrupt is asserted when the core detects an overflow or a CRC error For non-Isochronous OUT packet. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
6	back2backsetup	Back-to-Back SETUP Packets Received (Back2BackSETup) Applies to Control OUT endpoints only. This bit indicates that the core has received more than three back-to-back SETUP packets For this particular endpoint. For information about handling this interrupt, **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
5	stsphsercvd	Status Phase Received For Control Write (StsPhseRcvd) This interrupt is valid only For Control OUT endpoints and only in Scatter Gather DMA mode. This interrupt is generated only after the core has transferred all the data that the host has sent during the data phase of a control write transfer, to the system memory buffer. The interrupt indicates to the application that the host has switched from data phase to the status phase of a Control Write transfer. The application can use this interrupt to ACK or STALL the Status phase, after it has decoded the data phase. This is applicable only in Case of Scatter Gather DMA mode. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
4	outtknepdis	OUT Token Received When Endpoint Disabled (OUTTknEPdis) Applies only to control OUT endpoints. Indicates that an OUT token was received when the endpoint was not yet enabled. This interrupt is asserted on the endpoint For which the OUT token was received. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
3	setup	SETUP Phase Done (SetUp) Applies to control OUT endpoints only. Indicates that the SETUP phase For the control endpoint is complete and no more back-to-back SETUP packets were received For the current control transfer. On this interrupt, the application can decode the received SETUP data packet. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
2	ahberr	AHB Error (AHBErr) Applies to IN and OUT endpoints. This is generated only in Internal DMA mode when there is an AHB error during an AHB read/write. The application can read the corresponding endpoint DMA address register to get the error address. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
1	epdisbld	Endpoint Disabled Interrupt (EPDisbld) Applies to IN and OUT endpoints. This bit indicates that the endpoint is disabled per the application's request. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
0	xfercompl	Transfer Completed Interrupt (XferCompl) Applies to IN and OUT endpoints. When Scatter/Gather DMA mode is enabled - For IN endpoint this field indicates that the requested data from the descriptor is moved from external system memory to internal FIFO. - For OUT endpoint this field indicates that the requested data from the internal FIFO is moved to external system memory. This interrupt is generated only when the corresponding endpoint descriptor is closed, and the IOC bit For the corresponding descriptor is Set. When Scatter/Gather DMA mode is disabled, this field indicates that the programmed transfer is complete on the AHB as well as on the USB, For this endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

doeptsiz10

Device OUT Endpoint 10 Transfer Size Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00C50
i_usbotg_1_devgrp	0xFFB40800	0xFFB40C50

Offset: 0x450

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	rxdpid RO 0x0		pktcnt RW 0x0										xfersize RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
xfersize RW 0x0															

doeptsiz10 Fields

Bit	Name	Description	Access	Reset
30:29	rxdpid	Applies to isochronous OUT endpoints only. This is the data PID received in the last packet for this endpoint. 2'b00: DATA0 2'b01: DATA2 2'b10: DATA1 2'b11: MDATA SETUP Packet Count (SUPCnt) Applies to control OUT Endpoints only. This field specifies the number of back-to-back SETUP data packets the endpoint can receive. 2'b01: 1 packet 2'b10: 2 packets 2'b11: 3 packets **Value** **Description** 0x0 DATA0 0x1 DATA2PACKET1 0x2 DATA1PACKET2 0x3 MDATAPACKET3	RO	0x0
28:19	pktcnt	Packet Count (PktCnt) This field is decremented to zero after a packet is written into the RxFIFO.	RW	0x0
18:0	xfersize	Transfer Size (XferSize) Indicates the transfer size in bytes For endpoint 0. The core interrupts the application only after it has exhausted the transfer size amount of data. The transfer size can be Set to the maximum packet size of the endpoint, to be interrupted at the end of each packet. The core decrements this field every time a packet is read from the RxFIFO and written to the external memory.	RW	0x0

doepdma10

Device OUT Endpoint 10 DMA Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00C54
i_usbotg_1_devgrp	0xFFB40800	0xFFB40C54

Offset: 0x454

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdma10 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
doepdma10 RW 0x0															

doepdma10 Fields

Bit	Name	Description	Access	Reset
31:0	doepdma10	Holds the start address of the external memory for storing or fetching endpoint data. Note: For control endpoints, this field stores control OUT data packets as well as SETUP transaction data packets. When more than three SETUP packets are received back-to-back, the SETUP data packet in the memory is overwritten. This register is incremented on every AHB transaction. The application can give only a DWORD-aligned address. When Scatter/Gather DMA mode is not enabled, the application programs the start address value in this field. When Scatter/Gather DMA mode is enabled, this field indicates the base pointer for the descriptor list.	RW	0x0

doepdmab10

Device OUT Endpoint 10 Buffer Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00C5C
i_usbotg_1_devgrp	0xFFB40800	0xFFB40C5C

Offset: 0x45C

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdmab10 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
doepdmab10 RO 0x0															

doepdmab10 Fields

Bit	Name	Description	Access	Reset
31:0	doepdmab10	Holds the current buffer address.This register is updated as and when the data transfer for the corresponding end point is in progress. This register is present only in Scatter/Gather DMA mode. Otherwise this field is reserved.	RO	0x0

doepctl11

Device Control OUT Endpoint 11 Control Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00C60
i_usbotg_1_devgrp	0xFFB40800	0xFFB40C60

Offset: 0x460

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	Reserved				stall RW 0x0	snp RW 0x0	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
usbactep RW 0x0	Reserved				mps RW 0x0										

doepctl11 Fields

Bit	Name	Description	Access	Reset
31	epena	Endpoint Enable (EPEna) Applies to IN and OUT endpoints. When Scatter/Gather DMA mode is enabled, For IN endpoints this bit indicates that the descriptor structure and data buffer with data ready to transmit is setup. For OUT endpoint it indicates that the descriptor structure and data buffer to receive data is setup. When Scatter/Gather DMA mode is enabledsuch as for buffer-pointer based DMA mode: - For IN endpoints, this bit indicates that data is ready to be transmitted on the endpoint. - For OUT endpoints, this bit indicates that the application has allocated the memory to start receiving data from the USB. - The core clears this bit before setting any of the following interrupts on this endpoint: SETUP Phase Done Endpoint Disabled Transfer Completed Note: For control endpoints in DMA mode, this bit must be set to be able to transfer SETUP data packets in memory. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
30	epdis	Endpoint Disable (EPDis) Applies to IN and OUT endpoints. The application sets this bit to stop transmitting/receiving data on an endpoint, even before the transfer for that endpoint is complete. The application must wait for the Endpoint Disabled interrupt before treating the endpoint as disabled. The core clears this bit before setting the Endpoint Disabled interrupt. The application must set this bit only if Endpoint Enable is already set for this endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
29	setd1pid	Set DATA1 PID (SetD1PID) Applies to interrupt/bulk IN and OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA1. This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. Set Odd (micro)frame (SetOddFr) Applies to isochronous IN and OUT endpoints only. Writing to this field sets the Even/Odd (micro)frame (EO_FrNum) field to odd (micro)frame. This field is not applicable for Scatter/Gather DMA mode. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	WO	0x0

Bit	Name	Description	Access	Reset
28	setd0pid	Set DATA0 PID (SetD0PID) Applies to interrupt/bulk IN and OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA0. This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. 1'b0 WO In non-Scatter/Gather DMA mode: Set Even (micro)frame (SetEvenFr) Applies to isochronous IN and OUT endpoints only. Writing to this field sets the Even/Odd (micro)frame (EO_FrNum) field to even (micro) frame. When Scatter/Gather DMA mode is enabled, this field is reserved. The frame number in which to send data is in the transmit descriptor structure. The frame in which to receive data is updated in receive descriptor structure. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	WO	0x0
27	snak	Set NAK (SNAK) A write to this bit sets the NAK bit For the endpoint. Using this bit, the application can control the transmission of NAK handshakes on an endpoint. The core can also Set this bit For an endpoint after a SETUP packet is received on that endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	WO	0x0

Bit	Name	Description	Access	Reset
26	cnak	Clear NAK (CNAK) A write to this bit clears the NAK bit For the endpoint. Value: Description 0x0: INACTIVE 0x1: ACTIVE	WO	0x0
21	stall	STALL Handshake (Stall) Applies to non-control, non-isochronous IN and OUT endpoints only. The application sets this bit to stall all tokens from the USB host to this endpoint. If a NAK bit, Global Non-periodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority. Only the application can clear this bit, never the core. 1'b0 R_W Applies to control endpoints only. The application can only set this bit, and the core clears it, when a SETUP token is received for this endpoint. If a NAK bit, Global Non-periodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority. Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake. Value: Description 0x0: INACTIVE 0x1: ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
20	snp	Snoop Mode (Snp) Applies to OUT endpoints only. This bit configures the endpoint to Snoop mode. In Snoop mode, the core does not check the correctness of OUT packets before transferring them to application memory. **Value** **Description** 0x0 DISABLE 0x1 ENABLE	RW	0x0
19:18	eptype	Endpoint Type (EPType) This is the transfer type supported by this logical endpoint. 2'b00: Control 2'b01: Isochronous 2'b10: Bulk 2'b11: Interrupt **Value** **Description** 0x0 CONTROL 0x1 ISOCHRONOUS 0x2 BULK 0x3 INTERRUP	RW	0x0

Bit	Name	Description	Access	Reset
17	naksts	NAK Status (NAKSts) Indicates the following: 1'b0: The core is transmitting non-NAK handshakes based on the FIFO status. 1'b1: The core is transmitting NAK handshakes on this endpoint. When either the application or the core sets this bit: The core stops receiving any data on an OUT endpoint, even if there is space in the RxFIFO to accommodate the incoming packet. For non-isochronous IN endpoints: The core stops transmitting any data on an IN endpoint, even if there data is available in the TxFIFO. For isochronous IN endpoints: The core sends out a zero-length data packet, even if there data is available in the TxFIFO. Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake. **Value** — **Description** 0x0 — NONNAK 0x1 — NAK	RO	0x0

Bit	Name	Description	Access	Reset
16	dpid	Endpoint Data PID (DPID) Applies to interrupt/bulk IN and OUT endpoints only. Contains the PID of the packet to be received or transmitted on this endpoint. The application must program the PID of the first packet to be received or transmitted on this endpoint, after the endpoint is activated. The applications use the SetD1PID and SetD0PID fields of this register to program either DATA0 or DATA1 PID. 1'b0: DATA0 1'b1: DATA1 This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. 1'b0 RO Even/Odd (Micro)Frame (EO_FrNum) In non-Scatter/Gather DMA mode: Applies to isochronous IN and OUT endpoints only. Indicates the (micro)frame number in which the core transmits/ receives isochronous data for this endpoint. The application must program the even/odd (micro) frame number in which it intends to transmit/receive isochronous data for this endpoint using the SetEvnFr and SetOddFr fields in this register. 1'b0: Even (micro)frame 1'b1: Odd (micro)frame When Scatter/Gather DMA mode is enabled, this field is reserved. The frame number in which to send data is provided in the transmit descriptor structure. The frame in which data is received is updated in receive descriptor structure. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RO	0x0

Bit	Name	Description	Access	Reset
15	usbactep	USB Active Endpoint (USBActEP) Indicates whether this endpoint is active in the current configuration and interface. The core clears this bit for all endpoints (other than EP 0) after detecting a USB reset. After receiving the SetConfiguration and SetInterface commands, the applica-tion must program endpoint registers accordingly and set this bit. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
10:0	mps	Maximum Packet Size (MPS) The application must program this field with the maximum packet size for the current logical endpoint. This value is in bytes.	RW	0x0

doepint11

Device OUT Endpoint 11 Interrupt Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00C68
i_usbotg_1_devgrp	0xFFB40800	0xFFB40C68

Offset: 0x468

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
stuppktrcvd RW 0x0	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	outpkterr RW 0x0	Reserved	back2backsetup RW 0x0	stsphsercvd RW 0x0	outtknepdis RW 0x0	setup RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0

doepint11 Fields

Bit	Name	Description	Access	Reset
15	stuppktrcvd	Setup Packet Received Applicable for Control OUT Endpoints in only in the Buffer DMA Mode Set by the DWC_otg core, this bit indicates that this buffer holds 8 bytes of setup data. There is only one Setup packet per buffer. On receiving a Setup packet, the DWC_otg core closes the buffer and disables the corresponding endpoint. The applica-tion has to re-enable the endpoint to receive any OUT data for the Control Transfer and reprogram the buffer start address. Note: Because of the above behavior, the DWC_otg core can receive any number of back to back setup packets and one buffer for every setup packet is used. 1'b0: No Setup packet received 1'b1: Setup packet received Reset: 1’b0	RW	0x0

Bit	Name	Description	Access	Reset
14	nyetintrpt	NYET Interrupt (NYETIntrpt) The core generates this interrupt when a NYET response is transmitted for a non isochronous OUT endpoint. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
13	nakintrpt	NAK Interrupt (NAKInterrupt) The core generates this interrupt when a NAK is transmitted or received by the device. In case of isochronous IN endpoints the interrupt gets generated when a zero length packet is transmitted due to un-availability of data in the TXFifo. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
12	bbleerr	NAK Interrupt (BbleErr) The core generates this interrupt when babble is received for the endpoint. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
11	pktdrpsts	Packet Drop Status (PktDrpSts) This bit indicates to the application that an ISOC OUT packet has been dropped. This bit does not have an associated mask bit and does not generate an interrupt. Dependency: This bit is valid in non Scatter/Gather DMA mode when periodic transfer interrupt feature is selected. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
9	bnaintr	BNA (Buffer Not Available) Interrupt (BNAIntr) This bit is valid only when Scatter/Gather DMA mode is enabled. The core generates this interrupt when the descriptor accessed is not ready For the Core to process, such as Host busy or DMA done **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
8	outpkterr	OUT Packet Error (OutPktErr) Applies to OUT endpoints Only This interrupt is valid only when thresholding is enabled. This interrupt is asserted when the core detects an overflow or a CRC error For non-Isochronous OUT packet. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
6	back2backsetup	Back-to-Back SETUP Packets Received (Back2BackSETup) Applies to Control OUT endpoints only. This bit indicates that the core has received more than three back-to-back SETUP packets For this particular endpoint. For information about handling this interrupt, **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
5	stsphsercvd	Status Phase Received For Control Write (StsPhseRcvd) This interrupt is valid only For Control OUT endpoints and only in Scatter Gather DMA mode. This interrupt is generated only after the core has transferred all the data that the host has sent during the data phase of a control write transfer, to the system memory buffer. The interrupt indicates to the application that the host has switched from data phase to the status phase of a Control Write transfer. The application can use this interrupt to ACK or STALL the Status phase, after it has decoded the data phase. This is applicable only in Case of Scatter Gather DMA mode. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
4	outtknepdis	OUT Token Received When Endpoint Disabled (OUTTknEPdis) Applies only to control OUT endpoints. Indicates that an OUT token was received when the endpoint was not yet enabled. This interrupt is asserted on the endpoint For which the OUT token was received. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
3	setup	SETUP Phase Done (SetUp) Applies to control OUT endpoints only. Indicates that the SETUP phase For the control endpoint is complete and no more back-to-back SETUP packets were received For the current control transfer. On this interrupt, the application can decode the received SETUP data packet. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
2	ahberr	AHB Error (AHBErr) Applies to IN and OUT endpoints. This is generated only in Internal DMA mode when there is an AHB error during an AHB read/write. The application can read the corresponding endpoint DMA address register to get the error address. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
1	epdisbld	Endpoint Disabled Interrupt (EPDisbld) Applies to IN and OUT endpoints. This bit indicates that the endpoint is disabled per the application's request. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
0	xfercompl	Transfer Completed Interrupt (XferCompl) Applies to IN and OUT endpoints. When Scatter/Gather DMA mode is enabled - For IN endpoint this field indicates that the requested data from the descriptor is moved from external system memory to internal FIFO. - For OUT endpoint this field indicates that the requested data from the internal FIFO is moved to external system memory. This interrupt is generated only when the corresponding endpoint descriptor is closed, and the IOC bit For the corresponding descriptor is Set. When Scatter/Gather DMA mode is disabled, this field indicates that the programmed transfer is complete on the AHB as well as on the USB, For this endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

doeptsiz11

Device OUT Endpoint 11 Transfer Size Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00C70
i_usbotg_1_devgrp	0xFFB40800	0xFFB40C70

Offset: 0x470

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	rxdpid RO 0x0		pktcnt RW 0x0										xfersize RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
xfersize RW 0x0															

doeptsiz11 Fields

Bit	Name	Description	Access	Reset
30:29	rxdpid	Applies to isochronous OUT endpoints only. This is the data PID received in the last packet for this endpoint. 2'b00: DATA0 2'b01: DATA2 2'b10: DATA1 2'b11: MDATA SETUP Packet Count (SUPCnt) Applies to control OUT Endpoints only. This field specifies the number of back-to-back SETUP data packets the endpoint can receive. 2'b01: 1 packet 2'b10: 2 packets 2'b11: 3 packets **Value / Description** 0x0 DATA0 0x1 DATA2PACKET1 0x2 DATA1PACKET2 0x3 MDATAPACKET3	RO	0x0
28:19	pktcnt	Packet Count (PktCnt) This field is decremented to zero after a packet is written into the RxFIFO.	RW	0x0
18:0	xfersize	Transfer Size (XferSize) Indicates the transfer size in bytes For endpoint 0. The core interrupts the application only after it has exhausted the transfer size amount of data. The transfer size can be Set to the maximum packet size of the endpoint, to be interrupted at the end of each packet. The core decrements this field every time a packet is read from the RxFIFO and written to the external memory.	RW	0x0

doepdma11

Device OUT Endpoint 11 DMA Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00C74
i_usbotg_1_devgrp	0xFFB40800	0xFFB40C74

Offset: 0x474

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdma11 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
doepdma11 RW 0x0															

doepdma11 Fields

Bit	Name	Description	Access	Reset
31:0	doepdma11	Holds the start address of the external memory for storing or fetching endpoint data. Note: For control endpoints, this field stores control OUT data packets as well as SETUP transaction data packets. When more than three SETUP packets are received back-to-back, the SETUP data packet in the memory is overwritten. This register is incremented on every AHB transaction. The application can give only a DWORD-aligned address. When Scatter/Gather DMA mode is not enabled, the application programs the start address value in this field. When Scatter/Gather DMA mode is enabled, this field indicates the base pointer for the descriptor list.	RW	0x0

doepdmab11

Device OUT Endpoint 11 Buffer Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00C7C
i_usbotg_1_devgrp	0xFFB40800	0xFFB40C7C

Offset: 0x47C

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdmab11 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
doepdmab11 RO 0x0															

doepdmab11 Fields

Bit	Name	Description	Access	Reset
31:0	doepdmab11	Holds the current buffer address.This register is updated as and when the data transfer for the corresponding end point is in progress. This register is present only in Scatter/Gather DMA mode. Otherwise this field is reserved.	RO	0x0

doepctl12

Device Control OUT Endpoint 12 Control Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00C80
i_usbotg_1_devgrp	0xFFB40800	0xFFB40C80

Offset: 0x480

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	Reserved				stall RW 0x0	snp RW 0x0	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
usbactep RW 0x0	Reserved				mps RW 0x0										

doepctl12 Fields

Bit	Name	Description	Access	Reset
31	epena	Endpoint Enable (EPEna) Applies to IN and OUT endpoints. When Scatter/Gather DMA mode is enabled, For IN endpoints this bit indicates that the descriptor structure and data buffer with data ready to transmit is setup. For OUT endpoint it indicates that the descriptor structure and data buffer to receive data is setup. When Scatter/Gather DMA mode is enabledsuch as for buffer-pointer based DMA mode: - For IN endpoints, this bit indicates that data is ready to be transmitted on the endpoint. - For OUT endpoints, this bit indicates that the application has allocated the memory to start receiving data from the USB. - The core clears this bit before setting any of the following interrupts on this endpoint: SETUP Phase Done Endpoint Disabled Transfer Completed Note: For control endpoints in DMA mode, this bit must be set to be able to transfer SETUP data packets in memory. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
30	epdis	Endpoint Disable (EPDis) Applies to IN and OUT endpoints. The application sets this bit to stop transmitting/receiving data on an endpoint, even before the transfer for that endpoint is complete. The application must wait for the Endpoint Disabled interrupt before treating the endpoint as disabled. The core clears this bit before setting the Endpoint Disabled interrupt. The application must set this bit only if Endpoint Enable is already set for this endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
29	setd1pid	Set DATA1 PID (SetD1PID) Applies to interrupt/bulk IN and OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA1. This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. Set Odd (micro)frame (SetOddFr) Applies to isochronous IN and OUT endpoints only. Writing to this field sets the Even/Odd (micro)frame (EO_FrNum) field to odd (micro)frame. This field is not applicable for Scatter/Gather DMA mode. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	WO	0x0

Bit	Name	Description	Access	Reset
28	setd0pid	Set DATA0 PID (SetD0PID) Applies to interrupt/bulk IN and OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA0. This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. 1'b0 WO In non-Scatter/Gather DMA mode: Set Even (micro)frame (SetEvenFr) Applies to isochronous IN and OUT endpoints only. Writing to this field sets the Even/Odd (micro)frame (EO_FrNum) field to even (micro) frame. When Scatter/Gather DMA mode is enabled, this field is reserved. The frame number in which to send data is in the transmit descriptor structure. The frame in which to receive data is updated in receive descriptor structure. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	WO	0x0
27	snak	Set NAK (SNAK) A write to this bit sets the NAK bit For the endpoint. Using this bit, the application can control the transmission of NAK handshakes on an endpoint. The core can also Set this bit For an endpoint after a SETUP packet is received on that endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	WO	0x0

Bit	Name	Description	Access	Reset
26	cnak	Clear NAK (CNAK) A write to this bit clears the NAK bit For the endpoint. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	WO	0x0
21	stall	STALL Handshake (Stall) Applies to non-control, non-isochronous IN and OUT endpoints only. The application sets this bit to stall all tokens from the USB host to this endpoint. If a NAK bit, Global Non-periodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority. Only the application can clear this bit, never the core. 1'b0 R_W Applies to control endpoints only. The application can only set this bit, and the core clears it, when a SETUP token is received for this endpoint. If a NAK bit, Global Non-periodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority. Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
20	snp	Snoop Mode (Snp) Applies to OUT endpoints only. This bit configures the endpoint to Snoop mode. In Snoop mode, the core does not check the correctness of OUT packets before transferring them to application memory. **Value** **Description** 0x0 DISABLE 0x1 ENABLE	RW	0x0
19:18	eptype	Endpoint Type (EPType) This is the transfer type supported by this logical endpoint. 2'b00: Control 2'b01: Isochronous 2'b10: Bulk 2'b11: Interrupt **Value** **Description** 0x0 CONTROL 0x1 ISOCHRONOUS 0x2 BULK 0x3 INTERRUP	RW	0x0

Bit	Name	Description	Access	Reset
17	naksts	NAK Status (NAKSts) Indicates the following: 1'b0: The core is transmitting non-NAK handshakes based on the FIFO status. 1'b1: The core is transmitting NAK handshakes on this endpoint. When either the application or the core sets this bit: The core stops receiving any data on an OUT endpoint, even if there is space in the RxFIFO to accommodate the incoming packet. For non-isochronous IN endpoints: The core stops transmitting any data on an IN endpoint, even if there data is available in the TxFIFO. For isochronous IN endpoints: The core sends out a zero-length data packet, even if there data is available in the TxFIFO. Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake. **Value** — **Description** 0x0 — NONNAK 0x1 — NAK	RO	0x0

Bit	Name	Description	Access	Reset
16	dpid	Endpoint Data PID (DPID) Applies to interrupt/bulk IN and OUT endpoints only. Contains the PID of the packet to be received or transmitted on this endpoint. The application must program the PID of the first packet to be received or transmitted on this endpoint, after the endpoint is activated. The applications use the SetD1PID and SetD0PID fields of this register to program either DATA0 or DATA1 PID. 1'b0: DATA0 1'b1: DATA1 This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. 1'b0 RO Even/Odd (Micro)Frame (EO_FrNum) In non-Scatter/Gather DMA mode: Applies to isochronous IN and OUT endpoints only. Indicates the (micro)frame number in which the core transmits/receives isochronous data for this endpoint. The application must program the even/odd (micro) frame number in which it intends to transmit/receive isochronous data for this endpoint using the SetEvnFr and SetOddFr fields in this register. 1'b0: Even (micro)frame 1'b1: Odd (micro)frame When Scatter/Gather DMA mode is enabled, this field is reserved. The frame number in which to send data is provided in the transmit descriptor structure. The frame in which data is received is updated in receive descriptor structure. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RO	0x0

Bit	Name	Description	Access	Reset
15	usbactep	USB Active Endpoint (USBActEP) Indicates whether this endpoint is active in the current configuration and interface. The core clears this bit for all endpoints (other than EP 0) after detecting a USB reset. After receiving the SetConfiguration and SetInterface commands, the applica-tion must program endpoint registers accordingly and set this bit. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
10:0	mps	Maximum Packet Size (MPS) The application must program this field with the maximum packet size for the current logical endpoint. This value is in bytes.	RW	0x0

doepint12

Device OUT Endpoint 12 Interrupt Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00C88
i_usbotg_1_devgrp	0xFFB40800	0xFFB40C88

Offset: 0x488

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
stuppktrcvd RW 0x0	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	outpkterr RW 0x0	Reserved	back2backsetup RW 0x0	stsphsercvd RW 0x0	outtknepdis RW 0x0	setup RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0

doepint12 Fields

Bit	Name	Description	Access	Reset
15	stuppktrcvd	Setup Packet Received Applicable for Control OUT Endpoints in only in the Buffer DMA Mode Set by the DWC_otg core, this bit indicates that this buffer holds 8 bytes of setup data. There is only one Setup packet per buffer. On receiving a Setup packet, the DWC_otg core closes the buffer and disables the corresponding endpoint. The applica-tion has to re-enable the endpoint to receive any OUT data for the Control Transfer and reprogram the buffer start address. Note: Because of the above behavior, the DWC_otg core can receive any number of back to back setup packets and one buffer for every setup packet is used. 1'b0: No Setup packet received 1'b1: Setup packet received Reset: 1’b0	RW	0x0

Bit	Name	Description	Access	Reset
14	nyetintrpt	NYET Interrupt (NYETIntrpt) The core generates this interrupt when a NYET response is transmitted for a non isochronous OUT endpoint. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
13	nakintrpt	NAK Interrupt (NAKInterrupt) The core generates this interrupt when a NAK is transmitted or received by the device. In case of isochronous IN endpoints the interrupt gets generated when a zero length packet is transmitted due to un-availability of data in the TXFifo. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
12	bbleerr	NAK Interrupt (BbleErr) The core generates this interrupt when babble is received for the endpoint. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
11	pktdrpsts	Packet Drop Status (PktDrpSts) This bit indicates to the application that an ISOC OUT packet has been dropped. This bit does not have an associated mask bit and does not generate an interrupt. Dependency: This bit is valid in non Scatter/Gather DMA mode when periodic transfer interrupt feature is selected. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
9	bnaintr	BNA (Buffer Not Available) Interrupt (BNAIntr) This bit is valid only when Scatter/Gather DMA mode is enabled. The core generates this interrupt when the descriptor accessed is not ready For the Core to process, such as Host busy or DMA done **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
8	outpkterr	OUT Packet Error (OutPktErr) Applies to OUT endpoints Only This interrupt is valid only when thresholding is enabled. This interrupt is asserted when the core detects an overflow or a CRC error For non-Isochronous OUT packet. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
6	back2backsetup	Back-to-Back SETUP Packets Received (Back2BackSETup) Applies to Control OUT endpoints only. This bit indicates that the core has received more than three back-to-back SETUP packets For this particular endpoint. For information about handling this interrupt, **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
5	stsphsercvd	Status Phase Received For Control Write (StsPhseRcvd) This interrupt is valid only For Control OUT endpoints and only in Scatter Gather DMA mode. This interrupt is generated only after the core has transferred all the data that the host has sent during the data phase of a control write transfer, to the system memory buffer. The interrupt indicates to the application that the host has switched from data phase to the status phase of a Control Write transfer. The application can use this interrupt to ACK or STALL the Status phase, after it has decoded the data phase. This is applicable only in Case of Scatter Gather DMA mode. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
4	outtknepdis	OUT Token Received When Endpoint Disabled (OUTTknEPdis) Applies only to control OUT endpoints. Indicates that an OUT token was received when the endpoint was not yet enabled. This interrupt is asserted on the endpoint For which the OUT token was received. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
3	setup	SETUP Phase Done (SetUp) Applies to control OUT endpoints only. Indicates that the SETUP phase For the control endpoint is complete and no more back-to-back SETUP packets were received For the current control transfer. On this interrupt, the application can decode the received SETUP data packet. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
2	ahberr	AHB Error (AHBErr) Applies to IN and OUT endpoints. This is generated only in Internal DMA mode when there is an AHB error during an AHB read/write. The application can read the corresponding endpoint DMA address register to get the error address. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
1	epdisbld	Endpoint Disabled Interrupt (EPDisbld) Applies to IN and OUT endpoints. This bit indicates that the endpoint is disabled per the application's request. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
0	xfercompl	Transfer Completed Interrupt (XferCompl) Applies to IN and OUT endpoints. When Scatter/Gather DMA mode is enabled - For IN endpoint this field indicates that the requested data from the descriptor is moved from external system memory to internal FIFO. - For OUT endpoint this field indicates that the requested data from the internal FIFO is moved to external system memory. This interrupt is generated only when the corresponding endpoint descriptor is closed, and the IOC bit For the corresponding descriptor is Set. When Scatter/Gather DMA mode is disabled, this field indicates that the programmed transfer is complete on the AHB as well as on the USB, For this endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

doeptsiz12

Device OUT Endpoint 12 Transfer Size Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00C90
i_usbotg_1_devgrp	0xFFB40800	0xFFB40C90

Offset: 0x490

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	rxdpid RO 0x0		pktcnt RW 0x0										xfersize RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
xfersize RW 0x0															

doeptsiz12 Fields

Bit	Name	Description	Access	Reset
30:29	rxdpid	Applies to isochronous OUT endpoints only. This is the data PID received in the last packet for this endpoint. 2'b00: DATA0 2'b01: DATA2 2'b10: DATA1 2'b11: MDATA SETUP Packet Count (SUPCnt) Applies to control OUT Endpoints only. This field specifies the number of back-to-back SETUP data packets the endpoint can receive. 2'b01: 1 packet 2'b10: 2 packets 2'b11: 3 packets **Value** **Description** 0x0 DATA0 0x1 DATA2PACKET1 0x2 DATA1PACKET2 0x3 MDATAPACKET3	RO	0x0
28:19	pktcnt	Packet Count (PktCnt) This field is decremented to zero after a packet is written into the RxFIFO.	RW	0x0
18:0	xfersize	Transfer Size (XferSize) Indicates the transfer size in bytes For endpoint 0. The core interrupts the application only after it has exhausted the transfer size amount of data. The transfer size can be Set to the maximum packet size of the endpoint, to be interrupted at the end of each packet. The core decrements this field every time a packet is read from the RxFIFO and written to the external memory.	RW	0x0

doepdma12

Device OUT Endpoint 12 DMA Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00C94
i_usbotg_1_devgrp	0xFFB40800	0xFFB40C94

Offset: 0x494

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdma12 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
doepdma12 RW 0x0															

doepdma12 Fields

Bit	Name	Description	Access	Reset
31:0	doepdma12	Holds the start address of the external memory for storing or fetching endpoint data. Note: For control endpoints, this field stores control OUT data packets as well as SETUP transaction data packets. When more than three SETUP packets are received back-to-back, the SETUP data packet in the memory is overwritten. This register is incremented on every AHB transaction. The application can give only a DWORD-aligned address. When Scatter/Gather DMA mode is not enabled, the application programs the start address value in this field. When Scatter/Gather DMA mode is enabled, this field indicates the base pointer for the descriptor list.	RW	0x0

doepdmab12

Device OUT Endpoint 12 Buffer Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00C9C
i_usbotg_1_devgrp	0xFFB40800	0xFFB40C9C

Offset: 0x49C

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdmab12 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
doepdmab12 RO 0x0															

doepdmab12 Fields

Bit	Name	Description	Access	Reset
31:0	doepdmab12	Holds the current buffer address.This register is updated as and when the data transfer for the corresponding end point is in progress. This register is present only in Scatter/Gather DMA mode. Otherwise this field is reserved.	RO	0x0

doepctl13

Device Control OUT Endpoint 13 Control Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00CA0
i_usbotg_1_devgrp	0xFFB40800	0xFFB40CA0

Offset: 0x4A0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	Reserved				stall RW 0x0	snp RW 0x0	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
usbactep RW 0x0	Reserved				mps RW 0x0										

doepctl13 Fields

Bit	Name	Description	Access	Reset
31	epena	Endpoint Enable (EPEna) Applies to IN and OUT endpoints. When Scatter/Gather DMA mode is enabled, For IN endpoints this bit indicates that the descriptor structure and data buffer with data ready to transmit is setup. For OUT endpoint it indicates that the descriptor structure and data buffer to receive data is setup. When Scatter/Gather DMA mode is enabledsuch as for buffer-pointer based DMA mode: - For IN endpoints, this bit indicates that data is ready to be transmitted on the endpoint. - For OUT endpoints, this bit indicates that the application has allocated the memory to start receiving data from the USB. - The core clears this bit before setting any of the following interrupts on this endpoint: SETUP Phase Done Endpoint Disabled Transfer Completed Note: For control endpoints in DMA mode, this bit must be set to be able to transfer SETUP data packets in memory. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
30	epdis	Endpoint Disable (EPDis) Applies to IN and OUT endpoints. The application sets this bit to stop transmitting/receiving data on an endpoint, even before the transfer for that endpoint is complete. The application must wait for the Endpoint Disabled interrupt before treating the endpoint as disabled. The core clears this bit before setting the Endpoint Disabled interrupt. The application must set this bit only if Endpoint Enable is already set for this endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
29	setd1pid	Set DATA1 PID (SetD1PID) Applies to interrupt/bulk IN and OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA1. This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. Set Odd (micro)frame (SetOddFr) Applies to isochronous IN and OUT endpoints only. Writing to this field sets the Even/Odd (micro)frame (EO_FrNum) field to odd (micro)frame. This field is not applicable for Scatter/Gather DMA mode. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	WO	0x0

Bit	Name	Description	Access	Reset
28	setd0pid	Set DATA0 PID (SetD0PID) Applies to interrupt/bulk IN and OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA0. This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. 1'b0 WO In non-Scatter/Gather DMA mode: Set Even (micro)frame (SetEvenFr) Applies to isochronous IN and OUT endpoints only. Writing to this field sets the Even/Odd (micro)frame (EO_FrNum) field to even (micro) frame. When Scatter/Gather DMA mode is enabled, this field is reserved. The frame number in which to send data is in the transmit descriptor structure. The frame in which to receive data is updated in receive descriptor structure. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	WO	0x0
27	snak	Set NAK (SNAK) A write to this bit sets the NAK bit For the endpoint. Using this bit, the application can control the transmission of NAK handshakes on an endpoint. The core can also Set this bit For an endpoint after a SETUP packet is received on that endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	WO	0x0

Bit	Name	Description	Access	Reset
26	cnak	Clear NAK (CNAK) A write to this bit clears the NAK bit For the endpoint. Value: 0x0 — INACTIVE Value: 0x1 — ACTIVE	WO	0x0
21	stall	STALL Handshake (Stall) Applies to non-control, non-isochronous IN and OUT endpoints only. The application sets this bit to stall all tokens from the USB host to this endpoint. If a NAK bit, Global Non-periodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority. Only the application can clear this bit, never the core. 1'b0 R_W Applies to control endpoints only. The application can only set this bit, and the core clears it, when a SETUP token is received for this endpoint. If a NAK bit, Global Non-periodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority. Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake. Value: 0x0 — INACTIVE Value: 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
20	snp	Snoop Mode (Snp) Applies to OUT endpoints only. This bit configures the endpoint to Snoop mode. In Snoop mode, the core does not check the correctness of OUT packets before transferring them to application memory. **Value** **Description** 0x0 DISABLE 0x1 ENABLE	RW	0x0
19:18	eptype	Endpoint Type (EPType) This is the transfer type supported by this logical endpoint. 2'b00: Control 2'b01: Isochronous 2'b10: Bulk 2'b11: Interrupt **Value** **Description** 0x0 CONTROL 0x1 ISOCHRONOUS 0x2 BULK 0x3 INTERRUP	RW	0x0

Bit	Name	Description	Access	Reset
17	naksts	NAK Status (NAKSts) Indicates the following: 1'b0: The core is transmitting non-NAK handshakes based on the FIFO status. 1'b1: The core is transmitting NAK handshakes on this endpoint. When either the application or the core sets this bit: The core stops receiving any data on an OUT endpoint, even if there is space in the RxFIFO to accommodate the incoming packet. For non-isochronous IN endpoints: The core stops transmitting any data on an IN endpoint, even if there data is available in the TxFIFO. For isochronous IN endpoints: The core sends out a zero-length data packet, even if there data is available in the TxFIFO. Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake. **Value** — **Description** 0x0 — NONNAK 0x1 — NAK	RO	0x0

Bit	Name	Description	Access	Reset
16	dpid	Endpoint Data PID (DPID) Applies to interrupt/bulk IN and OUT endpoints only. Contains the PID of the packet to be received or transmitted on this endpoint. The application must program the PID of the first packet to be received or transmitted on this endpoint, after the endpoint is activated. The applications use the SetD1PID and SetD0PID fields of this register to program either DATA0 or DATA1 PID. 1'b0: DATA0 1'b1: DATA1 This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. 1'b0 RO Even/Odd (Micro)Frame (EO_FrNum) In non-Scatter/Gather DMA mode: Applies to isochronous IN and OUT endpoints only. Indicates the (micro)frame number in which the core transmits/ receives isochronous data for this endpoint. The application must program the even/odd (micro) frame number in which it intends to transmit/receive isochronous data for this endpoint using the SetEvnFr and SetOddFr fields in this register. 1'b0: Even (micro)frame 1'b1: Odd (micro)frame When Scatter/Gather DMA mode is enabled, this field is reserved. The frame number in which to send data is provided in the transmit descriptor structure. The frame in which data is received is updated in receive descriptor structure. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RO	0x0

Bit	Name	Description	Access	Reset
15	usbactep	USB Active Endpoint (USBActEP) Indicates whether this endpoint is active in the current configuration and interface. The core clears this bit for all endpoints (other than EP 0) after detecting a USB reset. After receiving the SetConfiguration and SetInterface commands, the applica-tion must program endpoint registers accordingly and set this bit. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
10:0	mps	Maximum Packet Size (MPS) The application must program this field with the maximum packet size for the current logical endpoint. This value is in bytes.	RW	0x0

doepint13

Device OUT Endpoint 13 Interrupt Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00CA8
i_usbotg_1_devgrp	0xFFB40800	0xFFB40CA8

Offset: 0x4A8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
stuppktrcvd RW 0x0	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	outpkterr RW 0x0	Reserved	back2backsetup RW 0x0	stsphsercvd RW 0x0	outtknepdis RW 0x0	setup RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0

doepint13 Fields

Bit	Name	Description	Access	Reset
15	stuppktrcvd	Setup Packet Received Applicable for Control OUT Endpoints in only in the Buffer DMA Mode Set by the DWC_otg core, this bit indicates that this buffer holds 8 bytes of setup data. There is only one Setup packet per buffer. On receiving a Setup packet, the DWC_otg core closes the buffer and disables the corresponding endpoint. The applica-tion has to re-enable the endpoint to receive any OUT data for the Control Transfer and reprogram the buffer start address. Note: Because of the above behavior, the DWC_otg core can receive any number of back to back setup packets and one buffer for every setup packet is used. 1'b0: No Setup packet received 1'b1: Setup packet received Reset: 1’b0	RW	0x0

Bit	Name	Description	Access	Reset
14	nyetintrpt	NYET Interrupt (NYETIntrpt) The core generates this interrupt when a NYET response is transmitted for a non isochronous OUT endpoint. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
13	nakintrpt	NAK Interrupt (NAKInterrupt) The core generates this interrupt when a NAK is transmitted or received by the device. In case of isochronous IN endpoints the interrupt gets generated when a zero length packet is transmitted due to un-availability of data in the TXFifo. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
12	bbleerr	NAK Interrupt (BbleErr) The core generates this interrupt when babble is received for the endpoint. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
11	pktdrpsts	Packet Drop Status (PktDrpSts) This bit indicates to the application that an ISOC OUT packet has been dropped. This bit does not have an associated mask bit and does not generate an interrupt. Dependency: This bit is valid in non Scatter/Gather DMA mode when periodic transfer interrupt feature is selected. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
9	bnaintr	BNA (Buffer Not Available) Interrupt (BNAIntr) This bit is valid only when Scatter/Gather DMA mode is enabled. The core generates this interrupt when the descriptor accessed is not ready For the Core to process, such as Host busy or DMA done **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
8	outpkterr	OUT Packet Error (OutPktErr) Applies to OUT endpoints Only This interrupt is valid only when thresholding is enabled. This interrupt is asserted when the core detects an overflow or a CRC error For non-Isochronous OUT packet. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
6	back2backsetup	Back-to-Back SETUP Packets Received (Back2BackSETup) Applies to Control OUT endpoints only. This bit indicates that the core has received more than three back-to-back SETUP packets For this particular endpoint. For information about handling this interrupt, **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
5	stsphsercvd	Status Phase Received For Control Write (StsPhseRcvd) This interrupt is valid only For Control OUT endpoints and only in Scatter Gather DMA mode. This interrupt is generated only after the core has transferred all the data that the host has sent during the data phase of a control write transfer, to the system memory buffer. The interrupt indicates to the application that the host has switched from data phase to the status phase of a Control Write transfer. The application can use this interrupt to ACK or STALL the Status phase, after it has decoded the data phase. This is applicable only in Case of Scatter Gather DMA mode. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
4	outtknepdis	OUT Token Received When Endpoint Disabled (OUTTknEPdis) Applies only to control OUT endpoints. Indicates that an OUT token was received when the endpoint was not yet enabled. This interrupt is asserted on the endpoint For which the OUT token was received. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
3	setup	SETUP Phase Done (SetUp) Applies to control OUT endpoints only. Indicates that the SETUP phase For the control endpoint is complete and no more back-to-back SETUP packets were received For the current control transfer. On this interrupt, the application can decode the received SETUP data packet. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
2	ahberr	AHB Error (AHBErr) Applies to IN and OUT endpoints. This is generated only in Internal DMA mode when there is an AHB error during an AHB read/write. The application can read the corresponding endpoint DMA address register to get the error address. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
1	epdisbld	Endpoint Disabled Interrupt (EPDisbld) Applies to IN and OUT endpoints. This bit indicates that the endpoint is disabled per the application's request. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
0	xfercompl	Transfer Completed Interrupt (XferCompl) Applies to IN and OUT endpoints. When Scatter/Gather DMA mode is enabled - For IN endpoint this field indicates that the requested data from the descriptor is moved from external system memory to internal FIFO. - For OUT endpoint this field indicates that the requested data from the internal FIFO is moved to external system memory. This interrupt is generated only when the corresponding endpoint descriptor is closed, and the IOC bit For the corresponding descriptor is Set. When Scatter/Gather DMA mode is disabled, this field indicates that the programmed transfer is complete on the AHB as well as on the USB, For this endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

doeptsiz13

Device OUT Endpoint 13 Transfer Size Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00CB0
i_usbotg_1_devgrp	0xFFB40800	0xFFB40CB0

Offset: 0x4B0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	rxdpid RO 0x0		pktcnt RW 0x0										xfersize RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
xfersize RW 0x0															

doeptsiz13 Fields

Bit	Name	Description	Access	Reset
30:29	rxdpid	Applies to isochronous OUT endpoints only. This is the data PID received in the last packet for this endpoint. 2'b00: DATA0 2'b01: DATA2 2'b10: DATA1 2'b11: MDATA SETUP Packet Count (SUPCnt) Applies to control OUT Endpoints only. This field specifies the number of back-to-back SETUP data packets the endpoint can receive. 2'b01: 1 packet 2'b10: 2 packets 2'b11: 3 packets **Value** **Description** 0x0 DATA0 0x1 DATA2PACKET1 0x2 DATA1PACKET2 0x3 MDATAPACKET3	RO	0x0
28:19	pktcnt	Packet Count (PktCnt) This field is decremented to zero after a packet is written into the RxFIFO.	RW	0x0
18:0	xfersize	Transfer Size (XferSize) Indicates the transfer size in bytes For endpoint 0. The core interrupts the application only after it has exhausted the transfer size amount of data. The transfer size can be Set to the maximum packet size of the endpoint, to be interrupted at the end of each packet. The core decrements this field every time a packet is read from the RxFIFO and written to the external memory.	RW	0x0

doepdma13

Device OUT Endpoint 13 DMA Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00CB4
i_usbotg_1_devgrp	0xFFB40800	0xFFB40CB4

Offset: 0x4B4

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdma13 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
doepdma13 RW 0x0															

doepdma13 Fields

Bit	Name	Description	Access	Reset
31:0	doepdma13	Holds the start address of the external memory for storing or fetching endpoint data. Note: For control endpoints, this field stores control OUT data packets as well as SETUP transaction data packets. When more than three SETUP packets are received back-to-back, the SETUP data packet in the memory is overwritten. This register is incremented on every AHB transaction. The application can give only a DWORD-aligned address. When Scatter/Gather DMA mode is not enabled, the application programs the start address value in this field. When Scatter/Gather DMA mode is enabled, this field indicates the base pointer for the descriptor list.	RW	0x0

doepdmab13

Device OUT Endpoint 13 Buffer Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00CBC
i_usbotg_1_devgrp	0xFFB40800	0xFFB40CBC

Offset: 0x4BC

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdmab13 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
doepdmab13 RO 0x0															

doepdmab13 Fields

Bit	Name	Description	Access	Reset
31:0	doepdmab13	Holds the current buffer address.This register is updated as and when the data transfer for the corresponding end point is in progress. This register is present only in Scatter/Gather DMA mode. Otherwise this field is reserved.	RO	0x0

doepctl14

Device Control OUT Endpoint 14 Control Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00CC0
i_usbotg_1_devgrp	0xFFB40800	0xFFB40CC0

Offset: 0x4C0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	Reserved				stall RW 0x0	snp RW 0x0	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
usbactep RW 0x0	Reserved				mps RW 0x0										

doepctl14 Fields

Bit	Name	Description	Access	Reset
31	epena	Endpoint Enable (EPEna) Applies to IN and OUT endpoints. When Scatter/Gather DMA mode is enabled, For IN endpoints this bit indicates that the descriptor structure and data buffer with data ready to transmit is setup. For OUT endpoint it indicates that the descriptor structure and data buffer to receive data is setup. When Scatter/Gather DMA mode is enabledsuch as for buffer-pointer based DMA mode: - For IN endpoints, this bit indicates that data is ready to be transmitted on the endpoint. - For OUT endpoints, this bit indicates that the application has allocated the memory to start receiving data from the USB. - The core clears this bit before setting any of the following interrupts on this endpoint: SETUP Phase Done Endpoint Disabled Transfer Completed Note: For control endpoints in DMA mode, this bit must be set to be able to transfer SETUP data packets in memory. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
30	epdis	Endpoint Disable (EPDis) Applies to IN and OUT endpoints. The application sets this bit to stop transmitting/receiving data on an endpoint, even before the transfer for that endpoint is complete. The application must wait for the Endpoint Disabled interrupt before treating the endpoint as disabled. The core clears this bit before setting the Endpoint Disabled interrupt. The application must set this bit only if Endpoint Enable is already set for this endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
29	setd1pid	Set DATA1 PID (SetD1PID) Applies to interrupt/bulk IN and OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA1. This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. Set Odd (micro)frame (SetOddFr) Applies to isochronous IN and OUT endpoints only. Writing to this field sets the Even/Odd (micro)frame (EO_FrNum) field to odd (micro)frame. This field is not applicable for Scatter/Gather DMA mode. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	WO	0x0

Bit	Name	Description	Access	Reset
28	setd0pid	Set DATA0 PID (SetD0PID) Applies to interrupt/bulk IN and OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA0. This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. 1'b0 WO In non-Scatter/Gather DMA mode: Set Even (micro)frame (SetEvenFr) Applies to isochronous IN and OUT endpoints only. Writing to this field sets the Even/Odd (micro)frame (EO_FrNum) field to even (micro) frame. When Scatter/Gather DMA mode is enabled, this field is reserved. The frame number in which to send data is in the transmit descriptor structure. The frame in which to receive data is updated in receive descriptor structure. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	WO	0x0
27	snak	Set NAK (SNAK) A write to this bit sets the NAK bit For the endpoint. Using this bit, the application can control the transmission of NAK handshakes on an endpoint. The core can also Set this bit For an endpoint after a SETUP packet is received on that endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	WO	0x0

Bit	Name	Description	Access	Reset
26	cnak	Clear NAK (CNAK) A write to this bit clears the NAK bit For the endpoint. Value: 0x0 — INACTIVE Value: 0x1 — ACTIVE	WO	0x0
21	stall	STALL Handshake (Stall) Applies to non-control, non-isochronous IN and OUT endpoints only. The application sets this bit to stall all tokens from the USB host to this endpoint. If a NAK bit, Global Non-periodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority. Only the application can clear this bit, never the core. 1'b0 R_W Applies to control endpoints only. The application can only set this bit, and the core clears it, when a SETUP token is received for this endpoint. If a NAK bit, Global Non-periodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority. Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake. Value: 0x0 — INACTIVE Value: 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
20	snp	Snoop Mode (Snp) Applies to OUT endpoints only. This bit configures the endpoint to Snoop mode. In Snoop mode, the core does not check the correctness of OUT packets before transferring them to application memory. **Value** — **Description** 0x0 — DISABLE 0x1 — ENABLE	RW	0x0
19:18	eptype	Endpoint Type (EPType) This is the transfer type supported by this logical endpoint. 2'b00: Control 2'b01: Isochronous 2'b10: Bulk 2'b11: Interrupt **Value** — **Description** 0x0 — CONTROL 0x1 — ISOCHRONOUS 0x2 — BULK 0x3 — INTERRUP	RW	0x0

Bit	Name	Description	Access	Reset
17	naksts	NAK Status (NAKSts) Indicates the following: 1'b0: The core is transmitting non-NAK handshakes based on the FIFO status. 1'b1: The core is transmitting NAK handshakes on this endpoint. When either the application or the core sets this bit: The core stops receiving any data on an OUT endpoint, even if there is space in the RxFIFO to accommodate the incoming packet. For non-isochronous IN endpoints: The core stops transmitting any data on an IN endpoint, even if there data is available in the TxFIFO. For isochronous IN endpoints: The core sends out a zero-length data packet, even if there data is available in the TxFIFO. Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake. **Value** — **Description** 0x0 — NONNAK 0x1 — NAK	RO	0x0

Bit	Name	Description	Access	Reset
16	dpid	Endpoint Data PID (DPID) Applies to interrupt/bulk IN and OUT endpoints only. Contains the PID of the packet to be received or transmitted on this endpoint. The application must program the PID of the first packet to be received or transmitted on this endpoint, after the endpoint is activated. The applications use the SetD1PID and SetD0PID fields of this register to program either DATA0 or DATA1 PID. 1'b0: DATA0 1'b1: DATA1 This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. 1'b0 RO Even/Odd (Micro)Frame (EO_FrNum) In non-Scatter/Gather DMA mode: Applies to isochronous IN and OUT endpoints only. Indicates the (micro)frame number in which the core transmits/receives isochronous data for this endpoint. The application must program the even/odd (micro) frame number in which it intends to transmit/receive isochronous data for this endpoint using the SetEvnFr and SetOddFr fields in this register. 1'b0: Even (micro)frame 1'b1: Odd (micro)frame When Scatter/Gather DMA mode is enabled, this field is reserved. The frame number in which to send data is provided in the transmit descriptor structure. The frame in which data is received is updated in receive descriptor structure. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RO	0x0

Bit	Name	Description	Access	Reset
15	usbactep	USB Active Endpoint (USBActEP) Indicates whether this endpoint is active in the current configuration and interface. The core clears this bit for all endpoints (other than EP 0) after detecting a USB reset. After receiving the SetConfiguration and SetInterface commands, the applica-tion must program endpoint registers accordingly and set this bit. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
10:0	mps	Maximum Packet Size (MPS) The application must program this field with the maximum packet size for the current logical endpoint. This value is in bytes.	RW	0x0

doepint14

Device OUT Endpoint 14 Interrupt Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00CC8
i_usbotg_1_devgrp	0xFFB40800	0xFFB40CC8

Offset: 0x4C8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
stuppktrcvd RW 0x0	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	outpkterr RW 0x0	Reserved	back2backsetup RW 0x0	stsphsercvd RW 0x0	outtknepdis RW 0x0	setup RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0

doepint14 Fields

Bit	Name	Description	Access	Reset
15	stuppktrcvd	Setup Packet Received Applicable for Control OUT Endpoints in only in the Buffer DMA Mode Set by the DWC_otg core, this bit indicates that this buffer holds 8 bytes of setup data. There is only one Setup packet per buffer. On receiving a Setup packet, the DWC_otg core closes the buffer and disables the corresponding endpoint. The applica-tion has to re-enable the endpoint to receive any OUT data for the Control Transfer and reprogram the buffer start address. Note: Because of the above behavior, the DWC_otg core can receive any number of back to back setup packets and one buffer for every setup packet is used. 1'b0: No Setup packet received 1'b1: Setup packet received Reset: 1’b0	RW	0x0

Bit	Name	Description	Access	Reset
14	nyetintrpt	NYET Interrupt (NYETIntrpt) The core generates this interrupt when a NYET response is transmitted for a non isochronous OUT endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
13	nakintrpt	NAK Interrupt (NAKInterrupt) The core generates this interrupt when a NAK is transmitted or received by the device. In case of isochronous IN endpoints the interrupt gets generated when a zero length packet is transmitted due to un-availability of data in the TXFifo. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
12	bbleerr	NAK Interrupt (BbleErr) The core generates this interrupt when babble is received for the endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
11	pktdrpsts	Packet Drop Status (PktDrpSts) This bit indicates to the application that an ISOC OUT packet has been dropped. This bit does not have an associated mask bit and does not generate an interrupt. Dependency: This bit is valid in non Scatter/Gather DMA mode when periodic transfer interrupt feature is selected. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
9	bnaintr	BNA (Buffer Not Available) Interrupt (BNAIntr) This bit is valid only when Scatter/Gather DMA mode is enabled. The core generates this interrupt when the descriptor accessed is not ready For the Core to process, such as Host busy or DMA done **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
8	outpkterr	OUT Packet Error (OutPktErr) Applies to OUT endpoints Only This interrupt is valid only when thresholding is enabled. This interrupt is asserted when the core detects an overflow or a CRC error For non-Isochronous OUT packet. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
6	back2backsetup	Back-to-Back SETUP Packets Received (Back2BackSETup) Applies to Control OUT endpoints only. This bit indicates that the core has received more than three back-to-back SETUP packets For this particular endpoint. For information about handling this interrupt, **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
5	stsphsercvd	Status Phase Received For Control Write (StsPhseRcvd) This interrupt is valid only For Control OUT endpoints and only in Scatter Gather DMA mode. This interrupt is generated only after the core has transferred all the data that the host has sent during the data phase of a control write transfer, to the system memory buffer. The interrupt indicates to the application that the host has switched from data phase to the status phase of a Control Write transfer. The application can use this interrupt to ACK or STALL the Status phase, after it has decoded the data phase. This is applicable only in Case of Scatter Gather DMA mode. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
4	outtknepdis	OUT Token Received When Endpoint Disabled (OUTTknEPdis) Applies only to control OUT endpoints. Indicates that an OUT token was received when the endpoint was not yet enabled. This interrupt is asserted on the endpoint For which the OUT token was received. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
3	setup	SETUP Phase Done (SetUp) Applies to control OUT endpoints only. Indicates that the SETUP phase For the control endpoint is complete and no more back-to-back SETUP packets were received For the current control transfer. On this interrupt, the application can decode the received SETUP data packet. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
2	ahberr	AHB Error (AHBErr) Applies to IN and OUT endpoints. This is generated only in Internal DMA mode when there is an AHB error during an AHB read/write. The application can read the corresponding endpoint DMA address register to get the error address. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
1	epdisbld	Endpoint Disabled Interrupt (EPDisbld) Applies to IN and OUT endpoints. This bit indicates that the endpoint is disabled per the application's request. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
0	xfercompl	Transfer Completed Interrupt (XferCompl) Applies to IN and OUT endpoints. When Scatter/Gather DMA mode is enabled - For IN endpoint this field indicates that the requested data from the descriptor is moved from external system memory to internal FIFO. - For OUT endpoint this field indicates that the requested data from the internal FIFO is moved to external system memory. This interrupt is generated only when the corresponding endpoint descriptor is closed, and the IOC bit For the corresponding descriptor is Set. When Scatter/Gather DMA mode is disabled, this field indicates that the programmed transfer is complete on the AHB as well as on the USB, For this endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

doeptsiz14

Device OUT Endpoint 14 Transfer Size Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00CD0
i_usbotg_1_devgrp	0xFFB40800	0xFFB40CD0

Offset: 0x4D0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	rxdpid RO 0x0		pktcnt RW 0x0										xfersize RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
xfersize RW 0x0															

doeptsiz14 Fields

Bit	Name	Description	Access	Reset
30:29	rxdpid	Applies to isochronous OUT endpoints only. This is the data PID received in the last packet for this endpoint. 2'b00: DATA0 2'b01: DATA2 2'b10: DATA1 2'b11: MDATA SETUP Packet Count (SUPCnt) Applies to control OUT Endpoints only. This field specifies the number of back-to-back SETUP data packets the endpoint can receive. 2'b01: 1 packet 2'b10: 2 packets 2'b11: 3 packets **Value** **Description** 0x0 DATA0 0x1 DATA2PACKET1 0x2 DATA1PACKET2 0x3 MDATAPACKET3	RO	0x0
28:19	pktcnt	Packet Count (PktCnt) This field is decremented to zero after a packet is written into the RxFIFO.	RW	0x0
18:0	xfersize	Transfer Size (XferSize) Indicates the transfer size in bytes For endpoint 0. The core interrupts the application only after it has exhausted the transfer size amount of data. The transfer size can be Set to the maximum packet size of the endpoint, to be interrupted at the end of each packet. The core decrements this field every time a packet is read from the RxFIFO and written to the external memory.	RW	0x0

doepdma14

Device OUT Endpoint 14 DMA Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00CD4
i_usbotg_1_devgrp	0xFFB40800	0xFFB40CD4

Offset: 0x4D4

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdma14 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
doepdma14 RW 0x0															

doepdma14 Fields

Bit	Name	Description	Access	Reset
31:0	doepdma14	Holds the start address of the external memory for storing or fetching endpoint data. Note: For control endpoints, this field stores control OUT data packets as well as SETUP transaction data packets. When more than three SETUP packets are received back-to-back, the SETUP data packet in the memory is overwritten. This register is incremented on every AHB transaction. The application can give only a DWORD-aligned address. When Scatter/Gather DMA mode is not enabled, the application programs the start address value in this field. When Scatter/Gather DMA mode is enabled, this field indicates the base pointer for the descriptor list.	RW	0x0

doepdmab14

Device OUT Endpoint 14 Buffer Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00CDC
i_usbotg_1_devgrp	0xFFB40800	0xFFB40CDC

Offset: 0x4DC

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdmab14 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
doepdmab14 RO 0x0															

doepdmab14 Fields

Bit	Name	Description	Access	Reset
31:0	doepdmab14	Holds the current buffer address.This register is updated as and when the data transfer for the corresponding end point is in progress. This register is present only in Scatter/Gather DMA mode. Otherwise this field is reserved.	RO	0x0

doepctl15

Device Control OUT Endpoint 15 Control Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00CE0
i_usbotg_1_devgrp	0xFFB40800	0xFFB40CE0

Offset: 0x4E0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
epena RW 0x0	epdis RW 0x0	setd1pid WO 0x0	setd0pid WO 0x0	snak WO 0x0	cnak WO 0x0	Reserved				stall RW 0x0	snp RW 0x0	eptype RW 0x0		naksts RO 0x0	dpid RO 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
usbactep RW 0x0	Reserved				mps RW 0x0										

doepctl15 Fields

Bit	Name	Description	Access	Reset
31	epena	Endpoint Enable (EPEna) Applies to IN and OUT endpoints. When Scatter/Gather DMA mode is enabled, For IN endpoints this bit indicates that the descriptor structure and data buffer with data ready to transmit is setup. For OUT endpoint it indicates that the descriptor structure and data buffer to receive data is setup. When Scatter/Gather DMA mode is enabledsuch as for buffer-pointer based DMA mode: - For IN endpoints, this bit indicates that data is ready to be transmitted on the endpoint. - For OUT endpoints, this bit indicates that the application has allocated the memory to start receiving data from the USB. - The core clears this bit before setting any of the following interrupts on this endpoint: SETUP Phase Done Endpoint Disabled Transfer Completed Note: For control endpoints in DMA mode, this bit must be set to be able to transfer SETUP data packets in memory. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
30	epdis	Endpoint Disable (EPDis) Applies to IN and OUT endpoints. The application sets this bit to stop transmitting/receiving data on an endpoint, even before the transfer for that endpoint is complete. The application must wait for the Endpoint Disabled interrupt before treating the endpoint as disabled. The core clears this bit before setting the Endpoint Disabled interrupt. The application must set this bit only if Endpoint Enable is already set for this endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
29	setd1pid	Set DATA1 PID (SetD1PID) Applies to interrupt/bulk IN and OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA1. This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. Set Odd (micro)frame (SetOddFr) Applies to isochronous IN and OUT endpoints only. Writing to this field sets the Even/Odd (micro)frame (EO_FrNum) field to odd (micro)frame. This field is not applicable for Scatter/Gather DMA mode. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	WO	0x0

Bit	Name	Description	Access	Reset
28	setd0pid	Set DATA0 PID (SetD0PID) Applies to interrupt/bulk IN and OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA0. This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. 1'b0 WO In non-Scatter/Gather DMA mode: Set Even (micro)frame (SetEvenFr) Applies to isochronous IN and OUT endpoints only. Writing to this field sets the Even/Odd (micro)frame (EO_FrNum) field to even (micro) frame. When Scatter/Gather DMA mode is enabled, this field is reserved. The frame number in which to send data is in the transmit descriptor structure. The frame in which to receive data is updated in receive descriptor structure. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	WO	0x0
27	snak	Set NAK (SNAK) A write to this bit sets the NAK bit For the endpoint. Using this bit, the application can control the transmission of NAK handshakes on an endpoint. The core can also Set this bit For an endpoint after a SETUP packet is received on that endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	WO	0x0

Bit	Name	Description	Access	Reset
26	cnak	Clear NAK (CNAK) A write to this bit clears the NAK bit For the endpoint. **Value** / **Description** 0x0 INACTIVE 0x1 ACTIVE	WO	0x0
21	stall	STALL Handshake (Stall) Applies to non-control, non-isochronous IN and OUT endpoints only. The application sets this bit to stall all tokens from the USB host to this endpoint. If a NAK bit, Global Non-periodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority. Only the application can clear this bit, never the core. 1'b0 R_W Applies to control endpoints only. The application can only set this bit, and the core clears it, when a SETUP token is received for this endpoint. If a NAK bit, Global Non-periodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority. Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake. **Value** / **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
20	snp	Snoop Mode (Snp) Applies to OUT endpoints only. This bit configures the endpoint to Snoop mode. In Snoop mode, the core does not check the correctness of OUT packets before transferring them to application memory. **Value** **Description** 0x0 DISABLE 0x1 ENABLE	RW	0x0
19:18	eptype	Endpoint Type (EPType) This is the transfer type supported by this logical endpoint. 2'b00: Control 2'b01: Isochronous 2'b10: Bulk 2'b11: Interrupt **Value** **Description** 0x0 CONTROL 0x1 ISOCHRONOUS 0x2 BULK 0x3 INTERRUP	RW	0x0

Bit	Name	Description	Access	Reset
17	naksts	NAK Status (NAKSts) Indicates the following: 1'b0: The core is transmitting non-NAK handshakes based on the FIFO status. 1'b1: The core is transmitting NAK handshakes on this endpoint. When either the application or the core sets this bit: The core stops receiving any data on an OUT endpoint, even if there is space in the RxFIFO to accommodate the incoming packet. For non-isochronous IN endpoints: The core stops transmitting any data on an IN endpoint, even if there data is available in the TxFIFO. For isochronous IN endpoints: The core sends out a zero-length data packet, even if there data is available in the TxFIFO. Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake. **Value** — **Description** 0x0 — NONNAK 0x1 — NAK	RO	0x0

Bit	Name	Description	Access	Reset
16	dpid	Endpoint Data PID (DPID) Applies to interrupt/bulk IN and OUT endpoints only. Contains the PID of the packet to be received or transmitted on this endpoint. The application must program the PID of the first packet to be received or transmitted on this endpoint, after the endpoint is activated. The applications use the SetD1PID and SetD0PID fields of this register to program either DATA0 or DATA1 PID. 1'b0: DATA0 1'b1: DATA1 This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. 1'b0 RO Even/Odd (Micro)Frame (EO_FrNum) In non-Scatter/Gather DMA mode: Applies to isochronous IN and OUT endpoints only. Indicates the (micro)frame number in which the core transmits/ receives isochronous data for this endpoint. The application must program the even/odd (micro) frame number in which it intends to transmit/receive isochronous data for this endpoint using the SetEvnFr and SetOddFr fields in this register. 1'b0: Even (micro)frame 1'b1: Odd (micro)frame When Scatter/Gather DMA mode is enabled, this field is reserved. The frame number in which to send data is provided in the transmit descriptor structure. The frame in which data is received is updated in receive descriptor structure. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RO	0x0

Bit	Name	Description	Access	Reset
15	usbactep	USB Active Endpoint (USBActEP) Indicates whether this endpoint is active in the current configuration and interface. The core clears this bit for all endpoints (other than EP 0) after detecting a USB reset. After receiving the SetConfiguration and SetInterface commands, the application must program endpoint registers accordingly and set this bit. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
10:0	mps	Maximum Packet Size (MPS) The application must program this field with the maximum packet size for the current logical endpoint. This value is in bytes.	RW	0x0

doepint15

Device OUT Endpoint 15 Interrupt Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00CE8
i_usbotg_1_devgrp	0xFFB40800	0xFFB40CE8

Offset: 0x4E8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
stuppktrcvd RW 0x0	nyetintrpt RW 0x0	nakintrpt RW 0x0	bbleerr RW 0x0	pktdrpsts RW 0x0	Reserved	bnaintr RW 0x0	outpkterr RW 0x0	Reserved	back2backsetup RW 0x0	stsphsercvd RW 0x0	outtknepdis RW 0x0	setup RW 0x0	ahberr RW 0x0	epdisbld RW 0x0	xfercompl RW 0x0

doepint15 Fields

Bit	Name	Description	Access	Reset
15	stuppktrcvd	Setup Packet Received Applicable for Control OUT Endpoints in only in the Buffer DMA Mode Set by the DWC_otg core, this bit indicates that this buffer holds 8 bytes of setup data. There is only one Setup packet per buffer. On receiving a Setup packet, the DWC_otg core closes the buffer and disables the corresponding endpoint. The applica-tion has to re-enable the endpoint to receive any OUT data for the Control Transfer and reprogram the buffer start address. Note: Because of the above behavior, the DWC_otg core can receive any number of back to back setup packets and one buffer for every setup packet is used. 1'b0: No Setup packet received 1'b1: Setup packet received Reset: 1’b0	RW	0x0

Bit	Name	Description	Access	Reset
14	nyetintrpt	NYET Interrupt (NYETIntrpt) The core generates this interrupt when a NYET response is transmitted for a non isochronous OUT endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
13	nakintrpt	NAK Interrupt (NAKInterrupt) The core generates this interrupt when a NAK is transmitted or received by the device. In case of isochronous IN endpoints the interrupt gets generated when a zero length packet is transmitted due to un-availability of data in the TXFifo. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
12	bbleerr	NAK Interrupt (BbleErr) The core generates this interrupt when babble is received for the endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
11	pktdrpsts	Packet Drop Status (PktDrpSts) This bit indicates to the application that an ISOC OUT packet has been dropped. This bit does not have an associated mask bit and does not generate an interrupt. Dependency: This bit is valid in non Scatter/Gather DMA mode when periodic transfer interrupt feature is selected. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
9	bnaintr	BNA (Buffer Not Available) Interrupt (BNAIntr) This bit is valid only when Scatter/Gather DMA mode is enabled. The core generates this interrupt when the descriptor accessed is not ready For the Core to process, such as Host busy or DMA done **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
8	outpkterr	OUT Packet Error (OutPktErr) Applies to OUT endpoints Only This interrupt is valid only when thresholding is enabled. This interrupt is asserted when the core detects an overflow or a CRC error For non-Isochronous OUT packet. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
6	back2backsetup	Back-to-Back SETUP Packets Received (Back2BackSETup) Applies to Control OUT endpoints only. This bit indicates that the core has received more than three back-to-back SETUP packets For this particular endpoint. For information about handling this interrupt, **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
5	stsphsercvd	Status Phase Received For Control Write (StsPhseRcvd) This interrupt is valid only For Control OUT endpoints and only in Scatter Gather DMA mode. This interrupt is generated only after the core has transferred all the data that the host has sent during the data phase of a control write transfer, to the system memory buffer. The interrupt indicates to the application that the host has switched from data phase to the status phase of a Control Write transfer. The application can use this interrupt to ACK or STALL the Status phase, after it has decoded the data phase. This is applicable only in Case of Scatter Gather DMA mode. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
4	outtknepdis	OUT Token Received When Endpoint Disabled (OUTTknEPdis) Applies only to control OUT endpoints. Indicates that an OUT token was received when the endpoint was not yet enabled. This interrupt is asserted on the endpoint For which the OUT token was received. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
3	setup	SETUP Phase Done (SetUp) Applies to control OUT endpoints only. Indicates that the SETUP phase For the control endpoint is complete and no more back-to-back SETUP packets were received For the current control transfer. On this interrupt, the application can decode the received SETUP data packet. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0
2	ahberr	AHB Error (AHBErr) Applies to IN and OUT endpoints. This is generated only in Internal DMA mode when there is an AHB error during an AHB read/write. The application can read the corresponding endpoint DMA address register to get the error address. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RW	0x0

Bit	Name	Description	Access	Reset
1	epdisbld	Endpoint Disabled Interrupt (EPDisbld) Applies to IN and OUT endpoints. This bit indicates that the endpoint is disabled per the application's request. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0
0	xfercompl	Transfer Completed Interrupt (XferCompl) Applies to IN and OUT endpoints. When Scatter/Gather DMA mode is enabled - For IN endpoint this field indicates that the requested data from the descriptor is moved from external system memory to internal FIFO. - For OUT endpoint this field indicates that the requested data from the internal FIFO is moved to external system memory. This interrupt is generated only when the corresponding endpoint descriptor is closed, and the IOC bit For the corresponding descriptor is Set. When Scatter/Gather DMA mode is disabled, this field indicates that the programmed transfer is complete on the AHB as well as on the USB, For this endpoint. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RW	0x0

doeptsiz15

Device OUT Endpoint 15 Transfer Size Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00CF0
i_usbotg_1_devgrp	0xFFB40800	0xFFB40CF0

Offset: 0x4F0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	rxdpid RO 0x0		pktcnt RW 0x0										xfersize RW 0x0		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
xfersize RW 0x0															

doeptsiz15 Fields

Bit	Name	Description	Access	Reset
30:29	rxdpid	Applies to isochronous OUT endpoints only. This is the data PID received in the last packet for this endpoint. 2'b00: DATA0 2'b01: DATA2 2'b10: DATA1 2'b11: MDATA SETUP Packet Count (SUPCnt) Applies to control OUT Endpoints only. This field specifies the number of back-to-back SETUP data packets the endpoint can receive. 2'b01: 1 packet 2'b10: 2 packets 2'b11: 3 packets **Value** **Description** 0x0 DATA0 0x1 DATA2PACKET1 0x2 DATA1PACKET2 0x3 MDATAPACKET3	RO	0x0
28:19	pktcnt	Packet Count (PktCnt) This field is decremented to zero after a packet is written into the RxFIFO.	RW	0x0
18:0	xfersize	Transfer Size (XferSize) Indicates the transfer size in bytes For endpoint 0. The core interrupts the application only after it has exhausted the transfer size amount of data. The transfer size can be Set to the maximum packet size of the endpoint, to be interrupted at the end of each packet. The core decrements this field every time a packet is read from the RxFIFO and written to the external memory.	RW	0x0

doepdma15

Device OUT Endpoint 15 DMA Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00CF4
i_usbotg_1_devgrp	0xFFB40800	0xFFB40CF4

Offset: 0x4F4

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdma15 RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
doepdma15 RW 0x0															

doepdma15 Fields

Bit	Name	Description	Access	Reset
31:0	doepdma15	Holds the start address of the external memory for storing or fetching endpoint data. Note: For control endpoints, this field stores control OUT data packets as well as SETUP transaction data packets. When more than three SETUP packets are received back-to-back, the SETUP data packet in the memory is overwritten. This register is incremented on every AHB transaction. The application can give only a DWORD-aligned address. When Scatter/Gather DMA mode is not enabled, the application programs the start address value in this field. When Scatter/Gather DMA mode is enabled, this field indicates the base pointer for the descriptor list.	RW	0x0

doepdmab15

Device OUT Endpoint 15 Buffer Address Register

Module Instance	Base Address	Register Address
i_usbotg_0_devgrp	0xFFB00800	0xFFB00CFC
i_usbotg_1_devgrp	0xFFB40800	0xFFB40CFC

Offset: 0x4FC

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
doepdmab15 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
doepdmab15 RO 0x0															

doepdmab15 Fields

Bit	Name	Description	Access	Reset
31:0	doepdmab15	Holds the current buffer address.This register is updated as and when the data transfer for the corresponding end point is in progress. This register is present only in Scatter/Gather DMA mode. Otherwise this field is reserved.	RO	0x0

usb_pwrclkgrp Address Map

Module Instance	Base Address	End Address
i_usbotg_0_pwrclkgrp	0xFFB00E00	0xFFB00FFF
i_usbotg_1_pwrclkgrp	0xFFB40E00	0xFFB40FFF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
pcgcctl on page 18-1394	0x0	32	RW	0x0	Power and Clock Gating Control Register

usb_pwrclkgrp Summary

Module Instance	Base Address
i_usbotg_0_pwrclkgrp	0xFFB00E00
i_usbotg_1_pwrclkgrp	0xFFB40E00

Register Address Offset	Bit Fields															
i_usbotg_0_pwrclkgrp																
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
pcgcctl 0x0	Reserved								l1suspended RO 0x0	physleep RO 0x0	Reserved		rstpdwnmodule RW 0x0	Reserved		stoppclk RW 0x0
i_usbotg_1_pwrclkgrp																
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
pcgcctl 0x0	Reserved								l1suspended RO 0x0	physleep RO 0x0	Reserved		rstpdwnmodule RW 0x0	Reserved		stoppclk RW 0x0

pcgcctl

Power and Clock Gating Control Register

Module Instance	Base Address	Register Address
i_usbotg_0_pwrclkgrp	0xFFB00E00	0xFFB00E00
i_usbotg_1_pwrclkgrp	0xFFB40E00	0xFFB40E00

Offset: 0x0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								l1suspended RO 0x0	physleep RO 0x0	Reserved		rstpdwnmodule RW 0x0	Reserved		stoppclk RW 0x0

pcgcctl Fields

Bit	Name	Description	Access	Reset
7	l1suspended	L1 Deep Sleep Indicates that the PHY is in deep sleep when in L1 state. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RO	0x0
6	physleep	PHY In Sleep Indicates that the PHY is in Sleep State. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RO	0x0

Bit	Name	Description	Access	Reset
3	rstpdwnmodule	Reset Power-Down Modules (RstPdwnModule) This bit is valid only in Partial Power-Down mode. The application sets this bit when the power is turned off. The application clears this bit after the power is turned on and the PHY clock is up.Note: The R/W of all core registers are possible only when this bit is set to 1b0. **Value** — **Description** 0x0 — ON 0x1 — OFF	RW	0x0
0	stoppclk	Stop Pclk (StopPclk) The application sets this bit to stop the PHY clock (phy_clk) when the USB is suspended, the session is not valid, or the device is disconnected. The application clears this bit when the USB is resumed or a new session starts. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0

usb_DWC_otg_DFIFO Address Map

Module Instance	Base Address	End Address
i_usbotg_0_DWC_otg_DFIFO_0	0xFFB01000	0xFFB01FFF
i_usbotg_0_DWC_otg_DFIFO_1	0xFFB02000	0xFFB02FFF
i_usbotg_0_DWC_otg_DFIFO_2	0xFFB03000	0xFFB03FFF

Module Instance	Base Address	End Address
i_usbotg_0_DWC_otg_DFIFO_3	0xFFB04000	0xFFB04FFF
i_usbotg_0_DWC_otg_DFIFO_4	0xFFB05000	0xFFB05FFF
i_usbotg_0_DWC_otg_DFIFO_5	0xFFB06000	0xFFB06FFF
i_usbotg_0_DWC_otg_DFIFO_6	0xFFB07000	0xFFB07FFF
i_usbotg_0_DWC_otg_DFIFO_7	0xFFB08000	0xFFB08FFF
i_usbotg_0_DWC_otg_DFIFO_8	0xFFB09000	0xFFB09FFF
i_usbotg_0_DWC_otg_DFIFO_9	0xFFB0A000	0xFFB0AFFF
i_usbotg_0_DWC_otg_DFIFO_10	0xFFB0B000	0xFFB0BFFF
i_usbotg_0_DWC_otg_DFIFO_11	0xFFB0C000	0xFFB0CFFF
i_usbotg_0_DWC_otg_DFIFO_12	0xFFB0D000	0xFFB0DFFF
i_usbotg_0_DWC_otg_DFIFO_13	0xFFB0E000	0xFFB0EFFF
i_usbotg_0_DWC_otg_DFIFO_14	0xFFB0F000	0xFFB0FFFF
i_usbotg_0_DWC_otg_DFIFO_15	0xFFB10000	0xFFB1FFFF
i_usbotg_1_DWC_otg_DFIFO_0	0xFFB41000	0xFFB41FFF
i_usbotg_1_DWC_otg_DFIFO_1	0xFFB42000	0xFFB42FFF
i_usbotg_1_DWC_otg_DFIFO_2	0xFFB43000	0xFFB43FFF
i_usbotg_1_DWC_otg_DFIFO_3	0xFFB44000	0xFFB44FFF
i_usbotg_1_DWC_otg_DFIFO_4	0xFFB45000	0xFFB45FFF
i_usbotg_1_DWC_otg_DFIFO_5	0xFFB46000	0xFFB46FFF

Module Instance	Base Address	End Address
i_usbotg_1_DWC_otg_DFIFO_6	0xFFB47000	0xFFB47FFF
i_usbotg_1_DWC_otg_DFIFO_7	0xFFB48000	0xFFB48FFF
i_usbotg_1_DWC_otg_DFIFO_8	0xFFB49000	0xFFB49FFF
i_usbotg_1_DWC_otg_DFIFO_9	0xFFB4A000	0xFFB4AFFF
i_usbotg_1_DWC_otg_DFIFO_10	0xFFB4B000	0xFFB4BFFF
i_usbotg_1_DWC_otg_DFIFO_11	0xFFB4C000	0xFFB4CFFF
i_usbotg_1_DWC_otg_DFIFO_12	0xFFB4D000	0xFFB4DFFF
i_usbotg_1_DWC_otg_DFIFO_13	0xFFB4E000	0xFFB4EFFF
i_usbotg_1_DWC_otg_DFIFO_14	0xFFB4F000	0xFFB4FFFF
i_usbotg_1_DWC_otg_DFIFO_15	0xFFB50000	0xFFB5FFFF

usb_DWC_otg_DFIFO_Direct_access Address Map

Module Instance	Base Address	End Address
i_usbotg_0_DWC_otg_DFIFO_Direct_access	0xFFB20000	0xFFB3FFFF
i_usbotg_1_DWC_otg_DFIFO_Direct_access	0xFFB60000	0xFFB7FFFF

Document Revision History

Table 18-4: Document Revision History

Date	Version	Changes
October 2016	2016.10.28	Maintenance release.

Date	Version	Changes
May 2016	2016.05.03	Maintenance release.
November 2015	2015.11.02	• Renamed "ULPI PHY Interface" section to "USB 2.0 ULPI PHY Signal Description" and moved it after the "USB OTG Controller Block Diagram and System Integration" section. • Removed references to LPM mode in document, including "LPM Function" section • Added "DMA" section • Added "**Clock Gating** on page 18-10" section
May 2015	2015.05.04	Maintenance release.
December 2014	2014.12.15	• Maintenance release. • Added *Taking the USB OTG Out of Reset* section.
August 2014	2014.08.18	Initial release.

SPI Controller 19

2016.10.28

a10_5v4 Subscribe Send Feedback

The hard processor system (HPS) provides two serial peripheral interface (SPI) masters and two SPI slaves. The SPI masters and slaves are instances of the Synopsys® DesignWare® Synchronous Serial Interface (SSI) controller (DW_apb_ssi). †[52]

Features of the SPI Controller

The SPI controller has the following features: †

- Serial master and serial slave controllers – Enable serial communication with serial-master or serial-slave peripheral devices. †
- Each SPI master has a maximum bit rate of 60Mbps
- Each SPI slave has a maximum bit rate of 50Mbps
- Serial interface operation – Programmable choice of the following protocols:
  - Motorola SPI protocol
  - Texas Instruments Synchronous Serial Protocol
  - National Semiconductor Microwire
- DMA controller interface integrated with HPS DMA controller
- SPI master supports received serial data bit (RXD) sample delay
- Transmit and receive FIFO buffers are 256 words deep
- SPI master supports up to four slave selects
- Programmable master serial bit rate
- Programmable data item size of 4 to 16 bits
- Support for Multi-master mode

[52] Portions © 2016 Synopsys, Inc. Used with permission. All rights reserved. Synopsys & DesignWare are registered trademarks of Synopsys, Inc. All documentation is provided "as is" and without any warranty. Synopsys expressly disclaims any and all warranties, express, implied, or otherwise, including the implied warranties of merchantability, fitness for a particular purpose, and non-infringement, and any warranties arising out of a course of dealing or usage of trade.

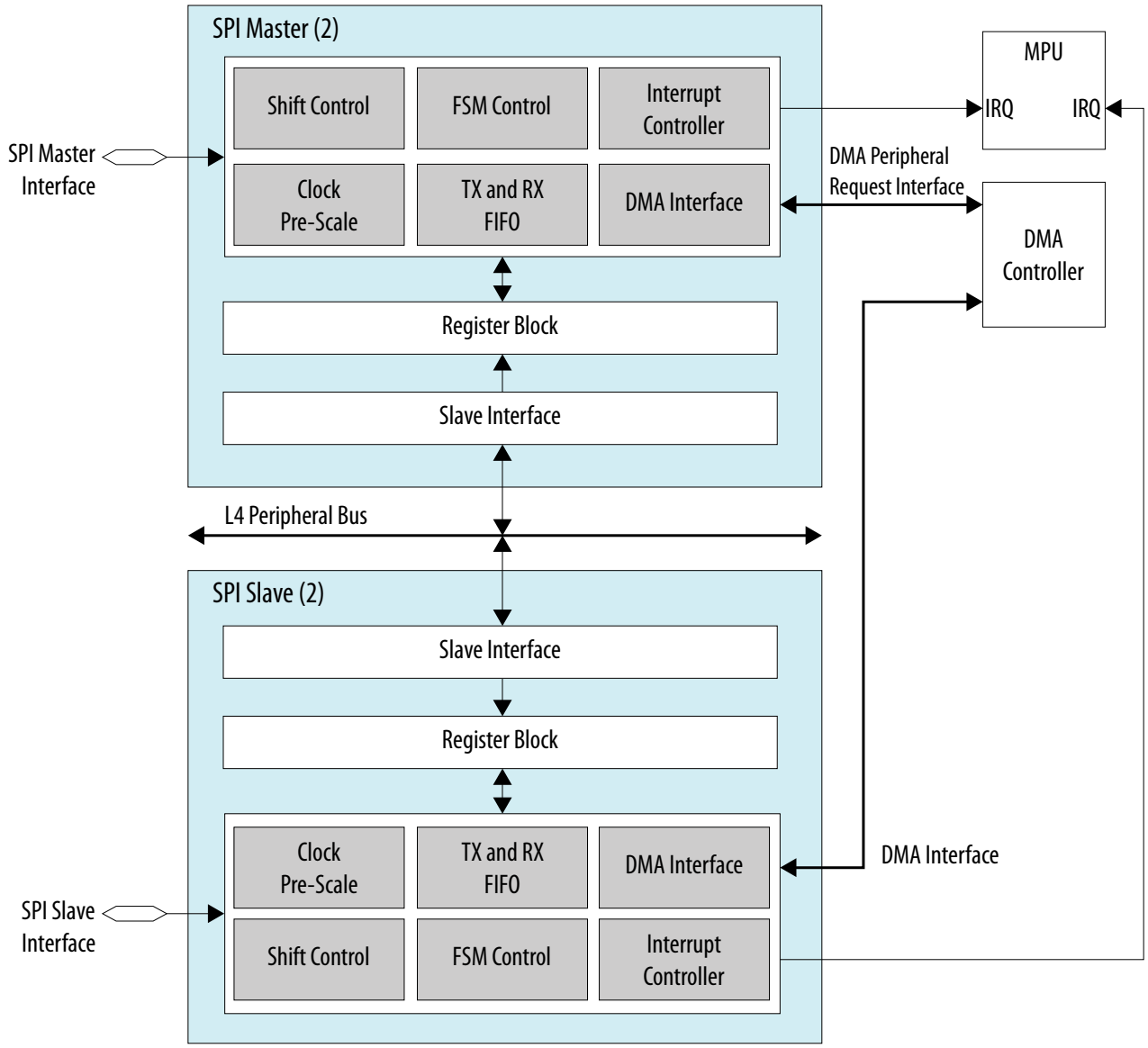
†Paragraphs marked with the dagger (†) symbol are Synopsys Proprietary. Used with permission.

SPI Block Diagram and System Integration

The SPI supports data bus widths of 32 bits. †

SPI Block Diagram

Figure 19-1: SPI Block Diagram



The functional groupings of the main interfaces to the SPI block are as follows: †

- System bus interface
- DMA peripheral request interface
- Interrupt interface
- SPI interface

SPI Controller Signal Description

Signals from the two SPI masters and two SPI slaves can be routed to the FPGA or the HPS I/O pins. The following sections describe the signals available.

Interface to HPS I/O

Two sets of SPI Master and two sets of SPI Slave Pins are available to the HPS I/O. The pin names are shown below. For more information on routing SPI signals to HPS I/O, refer to the *HPS Component Interfaces* chapter.

Table 19-1: SPI Master Interface Pins

Signal Name	Signal Width	Direction	Description
CLK	1	Out	Serial clock output from the SPI master
MOSI	1	Out	Transmit data line for the SPI master
MISO	1	In	Receive data line for the SPI master
SS0_N	1	Out	Slave Select 0: Slave select signal from SPI master
SS1_N	1	Out	Slave Select 1: Slave select signal from SPI master

Table 19-2: SPI Slave Interface Pins

Signal Name	Signal Width	Direction	Description
CLK	1	In	Serial clock input to the SPI slave
MOSI	1	In	Receive data line for the SPI slave
MISO	1	Out	Transmit data line for the SPI slave
SS0_N	1	In	Slave select input to the SPI slave

Related Information

HPS Component Interfaces on page 28-12
For information about routing SPI Controller signals, refer to this chapter.

FPGA Routing

Two sets of SPI Master and two sets of SPI Slave Pins are available for routing to the FPGA. The signal names are shown below. For more information on routing SPI signals to the FPGA, refer to the *HPS Component Interfaces* chapter.

Table 19-3: SPI Master Signals for FPGA Routing

Signal Name	Signal Width	Direction	Description
spim_mosi_o	1	Out	Transmit data line for the SPI master
spim_miso_i	1	In	Receive data line for the SPI master
spim_ss_in_n	1	In	Master Contention Input
spim_mosi_oe	1	Out	Output enable for the SPI master
spim_ss0_n_o	1	Out	Slave Select 0 Slave select signal from SPI master
spim_ss1_n_o	1	Out	Slave Select 1 Allows second slave to be connected to this master
spim_ss2_n_o	1	Out	Slave Select 2 Allows third slave to be connected to this master
spim_ss3_n_o	1	Out	Slave Select 3 Allows fourth slave to be connected to this master
spim_sclk_out	1	Out	Serial clock output

Table 19-4: SPI Slave Signals for FPGA Routing

Signal Name	Signal Width	Direction	Description
spis_miso_o	1	Out	Transmit data line for the SPI Slave
spis_mosi_i	1	In	Receive data line for the SPI Slave
spis_ss_in_n	1	Out	Master Contention Input
spis_miso_oe	1	Out	Output enable for the SPI Slave
spis_sclk_in	1	In	Serial clock input

Related Information

HPS Component Interfaces on page 28-12
For information about routing SPI Controller signals, refer to this chapter.

Functional Description of the SPI Controller

Protocol Details and Standards Compliance

This section describes the functional operation of the SPI controller.

The host processor accesses data, control, and status information about the SPI controller through the system bus interface. The SPI also interfaces with the DMA Controller. †

The HPS includes two general-purpose SPI master controllers and two general-purpose SPI slave controllers.

The SPI controller can connect to any other SPI device using any of the following protocols:

- Motorola SPI Protocol †
- Texas Instruments Serial Protocol (SSP) †
- National Semiconductor Microwire Protocol †

SPI Controller Overview

In order for the SPI controller to connect to a serial-master or serial-slave peripheral device, the peripheral must have a least one of the following interfaces: †

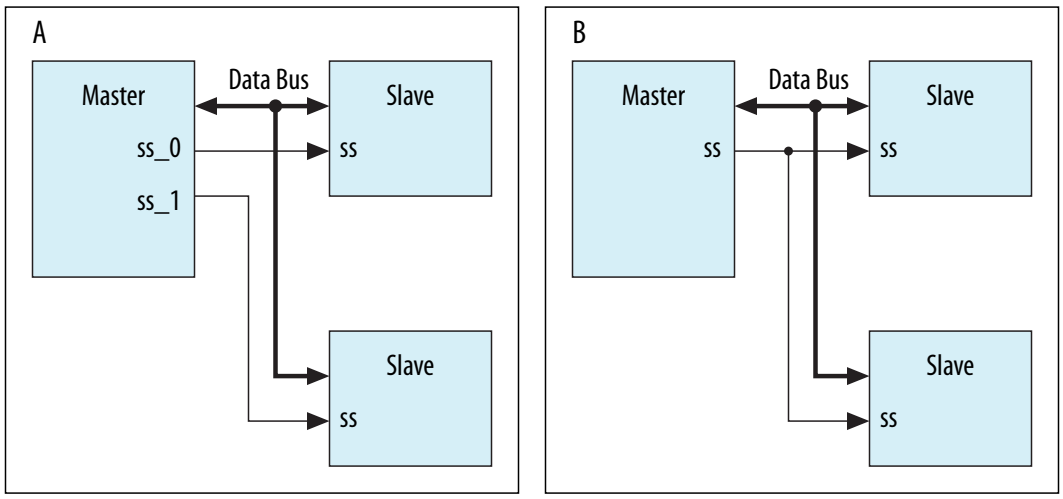
- Motorola SPI protocol – A four-wire, full-duplex serial protocol from Motorola. The slave select line is held high when the SPI controller is idle or disabled. For more information, refer to "Motorola SPI Protocol". †
- Texas Instruments Serial Protocol (SSP) – A four-wire, full-duplex serial protocol. The slave select line used for SPI and Microwire protocols doubles as the frame indicator for the SSP protocol. For more information, refer to "Texas Instruments Synchronous Serial Protocol (SSP)". †
- National Semiconductor Microwire – A half-duplex serial protocol, which uses a control word transmitted from the serial master to the target serial slave. For more information, refer to "National Semiconductor Microwire Protocol". You can program the FRF (frame format) bit field in the Control Register 0 (`CTRLR0`) to select which protocol is used. †

The serial protocols supported by the SPI controller allow for serial slaves to be selected or addressed using hardware. Serial slaves are selected under the control of dedicated hardware select lines. The number of select lines generated from the serial master is equal to the number of serial slaves present on the bus. The serial-master device asserts the select line of the target serial slave before data transfer begins. This architecture is illustrated in part A of **Figure 19-2**. †

When implemented in software, the input select line for all serial slave devices should originate from a single slave select output on the serial master. In this mode it is assumed that the serial master has only a single slave select output. If there are multiple serial masters in the system, the slave select output from all masters can be logically ANDed to generate a single slave select input for all serial slave devices. †

The main program in the software domain controls selection of the target slave device; this architecture is illustrated in part B of the **Figure 19-2** figure below. Software would control which slave is to respond to the serial transfer request from the master device. †

Figure 19-2: Hardware/Software Slave Selection



Related Information

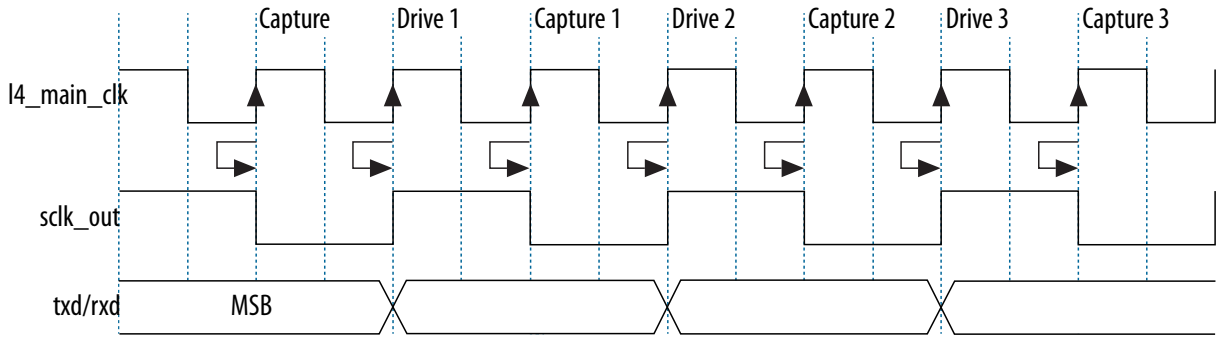
- **Motorola SPI Protocol** on page 19-16
- **Texas Instruments Synchronous Serial Protocol (SSP)** on page 19-18
- **National Semiconductor Microwire Protocol** on page 19-19

Serial Bit-Rate Clocks

SPI Master Bit-Rate Clock

The maximum frequency of the SPI master bit-rate clock (`sclk_out`) is one-half the frequency of SPI master clock (`l4_main_clk`). This allows the shift control logic to capture data on one clock edge of `sclk_out` and propagate data on the opposite edge. The `sclk_out` line toggles only when an active transfer is in progress. At all other times it is held in an inactive state, as defined by the serial protocol under which it operates. †

Figure 19-3: Maximum sclk_out/l4_main_clk Ratio



The frequency of `sclk_out` can be derived from the equation below, where <*SPI clock*> is `l4_main_clk` for both master and slave modules. †

Fsclk_out = F<*SPI clock*> / SCKDV

SCKDV is a bit field in the register BAUDR, holding any even value in the range 2 to 65,534. If SCKDV is 0, then sclk_out is disabled. †

The following equation describes the frequency ratio restrictions between the bit-rate clock sclk_out and the SPI master peripheral clock. The SPI master peripheral clock must be at least double the offchip master clock. †

Table 19-5: SPI Master Peripheral Clock

SPI Master Peripheral Clock
Fl4_main_clk >= 2 x (maximum $_{Fsclk\_out}$) †

SPI Slave Bit-Rate Clock

The minimum frequency of l4_main_clk depends on the operation of the slave peripheral. If the slave device is *receive only*, the minimum frequency of l4_main_clk is six times the maximum expected frequency of the bit-rate clock from the master device (sclk_in). The sclk_in signal is double synchronized to the l4_main_clk domain, and then it is edge detected; this synchronization requires three l4_main_clk periods. †

If the slave device is *transmit and receive*, the minimum frequency of l4_main_clk is eight times the maximum expected frequency of the bit-rate clock from the master device (sclk_in). This ensures that data on the master rxd line is stable before the master shift control logic captures the data. †

The frequency ratio restrictions between the bit-rate clock sclk_in and the SPI slave peripheral clock are as follows: †

- Slave (receive only): $F_{l4\_main\_clk}$ >= 6 multiply (maximum $_{Fsclk\_in}$) †
- Slave: $F_{l4\_main\_clk}$ >= 8 multiply (maximum F_{sclk_in}) †

Transmit and Receive FIFO Buffers

There are two 16-bit FIFO buffers, a transmit FIFO buffer and a receive FIFO buffer, with a depth of 256. Data frames that are less than 16 bits in size must be right-justified when written into the transmit FIFO buffer. The shift control logic automatically right-justifies receive data in the receive FIFO buffer. †

Each data entry in the FIFO buffers contains a single data frame. It is impossible to store multiple data frames in a single FIFO buffer location; for example, you may not store two 8-bit data frames in a single FIFO buffer location. If an 8-bit data frame is required, the upper 8-bits of the FIFO buffer entry are ignored or unused when the serial shifter transmits the data. †

The transmit and receive FIFO buffers are cleared when the SPI controller is disabled (SSIENR=0) or reset.

The transmit FIFO buffer is loaded by write commands to the SPI data register (DR). Data are popped (removed) from the transmit FIFO buffer by the shift control logic into the transmit shift register. The transmit FIFO buffer generates a transmit FIFO empty interrupt request when the number of entries in the FIFO buffer is less than or equal to the FIFO buffer threshold value. The threshold value, set through the register TXFTLR, determines the level of FIFO buffer entries at which an interrupt is generated. The threshold value allows you to provide early indication to the processor that the transmit FIFO buffer is nearly empty. A Transmit FIFO Overflow Interrupt is generated if you attempt to write data into an already full transmit FIFO buffer. †

Data are popped from the receive FIFO buffer by read commands to the SPI data register (DR). The receive FIFO buffer is loaded from the receive shift register by the shift control logic. The receive FIFO buffer generates a receive FIFO full interrupt request when the number of entries in the FIFO buffer is greater than or equal to the FIFO buffer threshold value plus one. The threshold value, set through register RXFTLR, determines the level of FIFO buffer entries at which an interrupt is generated. †

The threshold value allows you to provide early indication to the processor that the receive FIFO buffer is nearly full. A Receive FIFO Overrun Interrupt is generated when the receive shift logic attempts to load data into a completely full receive FIFO buffer. However, the newly received data are lost. A Receive FIFO Underflow Interrupt is generated if you attempt to read from an empty receive FIFO buffer. This alerts the processor that the read data are invalid. †

Related Information

Reset Manager on page 3-1
For more information, refer to the*Reset Manager* chapter.

SPI Interrupts

The SPI controller supports combined interrupt requests, which can be masked. The combined interrupt request is the ORed result of all other SPI interrupts after masking. All SPI interrupts have active-high polarity level. The SPI interrupts are described as follows: †

- Transmit FIFO Empty Interrupt – Set when the transmit FIFO buffer is equal to or below its threshold value and requires service to prevent an underrun. The threshold value, set through a software-programmable register, determines the level of transmit FIFO buffer entries at which an interrupt is generated. This interrupt is cleared by hardware when data are written into the transmit FIFO buffer, bringing it over the threshold level. †
- Transmit FIFO Overflow Interrupt – Set when a master attempts to write data into the transmit FIFO buffer after it has been completely filled. When set, new data writes are discarded. This interrupt remains set until you read the transmit FIFO overflow interrupt clear register (TXOICR). †
- Receive FIFO Full Interrupt – Set when the receive FIFO buffer is equal to or above its threshold value plus 1 and requires service to prevent an overflow. The threshold value, set through a software-programmable register, determines the level of receive FIFO buffer entries at which an interrupt is generated. This interrupt is cleared by hardware when data are read from the receive FIFO buffer, bringing it below the threshold level. †
- Receive FIFO Overflow Interrupt – Set when the receive logic attempts to place data into the receive FIFO buffer after it has been completely filled. When set, newly received data are discarded. This interrupt remains set until you read the receive FIFO overflow interrupt clear register (RXOICR). †
- Receive FIFO Underflow Interrupt – Set when a system bus access attempts to read from the receive FIFO buffer when it is empty. When set, zeros are read back from the receive FIFO buffer. This interrupt remains set until you read the receive FIFO underflow interrupt clear register (RXUICR). †
- Combined Interrupt Request – ORed result of all the above interrupt requests after masking. To mask this interrupt signal, you must mask all other SPI interrupt requests. †

Transmit FIFO Overflow, Transmit FIFO Empty, Receive FIFO Full, Receive FIFO Underflow, and Receive FIFO Overflow interrupts can all be masked independently, using the Interrupt Mask Register (IMR). †

Transfer Modes

When transferring data on the serial bus, the SPI controller operates one of several modes. The transfer mode (TMOD) is set by writing to the TMOD field in control register 0 (CTRLR0).

Note: The transfer mode setting does not affect the duplex of the serial transfer. TMOD is ignored for Microwire transfers, which are controlled by the MWCR register. †

Transmit and Receive

When TMOD = 0, both transmit and receive logic are valid. The data transfer occurs as normal according to the selected frame format (serial protocol). Transmit data are popped from the transmit FIFO buffer and sent through the `txd` line to the target device, which replies with data on the `rxd` line. The receive data from the target device is moved from the receive shift register into the receive FIFO buffer at the end of each data frame. †

Transmit Only

When TMOD = 1, any receive data are ignored. The data transfer occurs as normal, according to the selected frame format (serial protocol). Transmit data are popped from the transmit FIFO buffer and sent through the `txd` line to the target device, which replies with data on the `rxd` line. At the end of the data frame, the receive shift register does not load its newly received data into the receive FIFO buffer. The data in the receive shift register is overwritten by the next transfer. You should mask interrupts originating from the receive logic when this mode is entered. †

Receive Only

When TMOD = 2, the transmit data are invalid. In the case of the SPI slave, the transmit FIFO buffer is never popped in Receive Only mode. The `txd` output remains at a constant logic level during the transmission. The data transfer occurs as normal according to the selected frame format (serial protocol). The receive data from the target device is moved from the receive shift register into the receive FIFO buffer at the end of each data frame. You should mask interrupts originating from the transmit logic when this mode is entered. †

EEPROM Read

Note: This transfer mode is only valid for serial masters. †

When TMOD = 3, the transmit data is used to transmit an opcode and/or an address to the EEPROM device. This takes three data frames (8-bit opcode followed by 8-bit upper address and 8-bit lower address). During the transmission of the opcode and address, no data is captured by the receive logic (as long as the SPI master is transmitting data on its `txd` line, data on the `rxd` line is ignored). The SPI master continues to transmit data until the transmit FIFO buffer is empty. You should ONLY have enough data frames in the transmit FIFO buffer to supply the opcode and address to the EEPROM. If more data frames are in the transmit FIFO buffer than are needed, then Read data is lost. †

When the transmit FIFO buffer becomes empty (all control information has been sent), data on the receive line (`rxd`) is valid and is stored in the receive FIFO buffer; the `txd` output is held at a constant logic level. The serial transfer continues until the number of data frames received by the SPI master matches the value of the NDF field in the `CTRLR1` register plus one. †

Note: EEPROM read mode is not supported when the SPI controller is configured to be in the SSP mode. †

SPI Master

The SPI master initiates and controls all serial transfers with serial-slave peripheral devices. †

The serial bit-rate clock, generated and controlled by the SPI controller, is driven out on the `sclk_out` line. When the SPI controller is disabled, no serial transfers can occur and `sclk_out` is held in "inactive" state, as defined by the serial protocol under which it operates. †

Related Information

SPI Block Diagram on page 19-2

RXD Sample Delay

The SPI master device is capable of delaying the default sample time of the `rxd` signal in order to increase the maximum achievable frequency on the serial bus.

Round trip routing delays on the `sclk_out` signal from the master and the `rxd` signal from the slave can mean that the timing of the `rxd` signal, as seen by the master, has moved away from the normal sampling time.

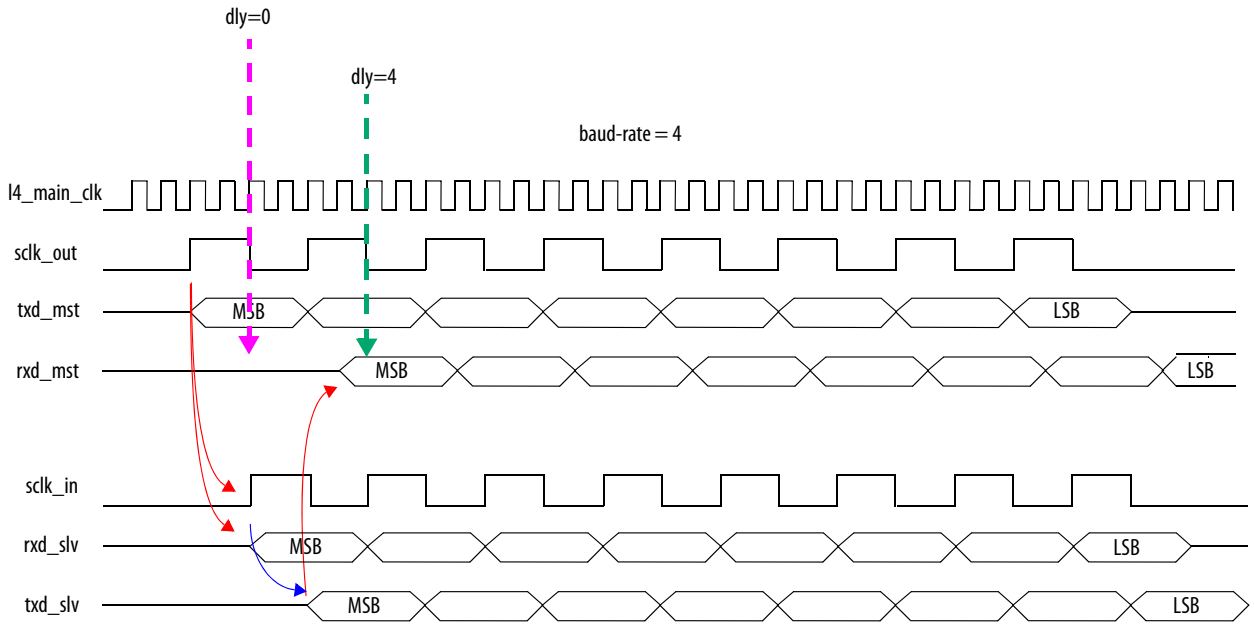
Without the RXD sample delay, you must increase the baud rate for the transfer in order to ensure that the setup times on the `rxd` signal are within range. This reduces the frequency of the serial interface.

Additional logic is included in the SPI master to delay the default sample time of the `rxd` signal. This additional logic can help to increase the maximum achievable frequency on the serial bus. †

By writing to the `rsd` field of the RXD sample delay register (`rx_sample_dly`), you specify an additional amount of delay applied to the `rxd` sample. The delay is in number of `l4_main_clk` clock cycles, with 64 maximum cycles allowed (zero is reserved). If the `rsd` field is programmed with a value exceeding 64, a zero delay is applied to the `rxd` sample.

The sample delay logic has a resolution of one `l4_main_clk` cycle. Software can "train" the serial bus by coding a loop that continually reads from the slave and increments the master's RXD sample delay value until the correct data is received by the master. †

Figure 19-4: Effects of Round Trip Routing Delays on sclk_out Signal



Red arrows indicates routing delay between master and slave devices
Blue arrow indicates sampling delay within slave from receiving slk_in to driving txd out

Data Transfers

The SPI master starts data transfers when all the following conditions are met:

- The SPI master is enabled
- There is at least one valid entry in the transmit FIFO buffer
- A slave device is selected

When actively transferring data, the busy flag (BUSY) in the status register (SR) is set. You must wait until the busy flag is cleared before attempting a new serial transfer. †

Note: The BUSY status is not set when the data are written into the transmit FIFO buffer. This bit gets set only when the target slave has been selected and the transfer is underway. After writing data into the transmit FIFO buffer, the shift logic does not begin the serial transfer until a positive edge of the sclk_out signal is present. The delay in waiting for this positive edge depends on the baud rate of the serial transfer. Before polling the BUSY status, you should first poll the Transit FIFO Empty (TFE) status (waiting for 1) or wait for (BAUDR * SPI clock) clock cycles. †

Master SPI and SSP Serial Transfers

"Motorola SPI Protocol" and "Texas Instruments Synchronous Serial Protocol (SSP)" describe the SPI and SSP serial protocols, respectively. †

When the transfer mode is "transmit and receive" or "transmit only" (TMOD = 0 or TMOD = 1, respectively), transfers are terminated by the shift control logic when the transmit FIFO buffer is empty. For continuous data transfers, you must ensure that the transmit FIFO buffer does not become empty before all the data have been transmitted. The transmit FIFO threshold level (TXFTLR) can be used to early interrupt (Transmit FIFO Empty Interrupt) the processor indicating that the transmit FIFO buffer is nearly empty. †

When the DMA is used in conjunction with the SPI master, the transmit data level (DMATDLR) can be used to early request the DMA Controller, indicating that the transmit FIFO buffer is nearly empty. The FIFO buffer can then be refilled with data to continue the serial transfer. The user may also write a block of data (at least two FIFO buffer entries) into the transmit FIFO buffer before enabling a serial slave. This ensures that serial transmission does not begin until the number of data frames that make up the continuous transfer are present in the transmit FIFO buffer. †

When the transfer mode is "receive only" (TMOD = 2), a serial transfer is started by writing one "dummy" data word into the transmit FIFO buffer when a serial slave is selected. The txd output from the SPI controller is held at a constant logic level for the duration of the serial transfer. The transmit FIFO buffer is popped only once at the beginning and may remain empty for the duration of the serial transfer. The end of the serial transfer is controlled by the "number of data frames" (NDF) field in control register 1 (CTRLR1). †

If, for example, you want to receive 24 data frames from a serial-slave peripheral, you should program the NDF field with the value 23; the receive logic terminates the serial transfer when the number of frames received is equal to the NDF value plus one. This transfer mode increases the bandwidth of the system bus as the transmit FIFO buffer never needs to be serviced during the transfer. The receive FIFO buffer should be read each time the receive FIFO buffer generates a FIFO full interrupt request to prevent an overflow. †

When the transfer mode is "eeprom_read" (TMOD = 3), a serial transfer is started by writing the opcode and/or address into the transmit FIFO buffer when a serial slave (EEPROM) is selected. The opcode and address are transmitted to the EEPROM device, after which read data is received from the EEPROM device and stored in the receive FIFO buffer. The end of the serial transfer is controlled by the NDF field in the control register 1 (CTRLR1). †

Note: EEPROM read mode is not supported when the SPI controller is configured to be in the SSP mode. †

The receive FIFO threshold level (RXF TLR) can be used to give early indication that the receive FIFO buffer is nearly full. When a DMA is used, the receive data level (DMARDLR) can be used to early request the DMA Controller, indicating that the receive FIFO buffer is nearly full. †

Related Information

- **Motorola SPI Protocol** on page 19-16
- **Texas Instruments Synchronous Serial Protocol (SSP)** on page 19-18

Master Microwire Serial Transfers

"National Semiconductor Microwire Protocol" describes the Microwire serial protocol in detail. †

Microwire serial transfers from the SPI serial master are controlled by the Microwire Control Register (MWCR). The MHS bit field enables and disables the Microwire handshaking interface. The MDD bit field controls the direction of the data frame (the control frame is always transmitted by the master and received by the slave). The MWMOD bit field defines whether the transfer is sequential or nonsequential. †

All Microwire transfers are started by the SPI serial master when there is at least one control word in the transmit FIFO buffer and a slave is enabled. When the SPI master transmits the data frame (MDD =1), the transfer is terminated by the shift logic when the transmit FIFO buffer is empty. When the SPI master receives the data frame (MDD = 1), the termination of the transfer depends on the setting of the MWMOD bit field. If the transfer is nonsequential (MWMOD = 0), it is terminated when the transmit FIFO buffer is empty after shifting in the data frame from the slave. When the transfer is sequential (MWMOD = 1), it is terminated by the shift logic when the number of data frames received is equal to the value in the CTRLR1 register plus one. †

When the handshaking interface on the SPI master is enabled (MHS =1), the status of the target slave is polled after transmission. Only when the slave reports a *ready status* does the SPI master complete the transfer and clear its BUSY status. If the transfer is continuous, the next control/data frame is not sent until the slave device returns a *ready status*. †

Related Information

National Semiconductor Microwire Protocol on page 19-19

SPI Slave

The SPI slave handles serial communication with transfer initiated and controlled by serial master peripheral devices.

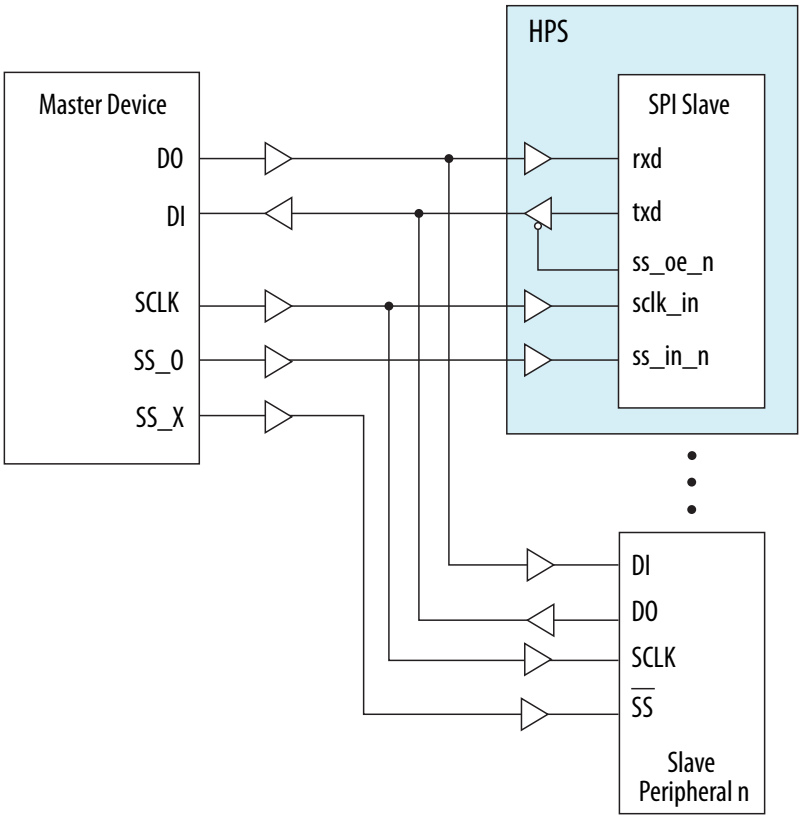
- sclk_in—serial clock to the SPI slave †
- ss_in_n—slave select input to the SPI slave †
- ss_oe_n—output enable for the SPI master or slave †
- txd—transmit data line for the SPI master or slave †
- rxd—receive data line for the SPI master or slave †

When the SPI serial slave is selected, it enables its txd data onto the serial bus. All data transfers to and from the serial slave are regulated on the serial clock line (sclk_in), driven from the SPI master device. Data are propagated from the serial slave on one edge of the serial clock line and sampled on the opposite edge. †

When the SPI serial slave is not selected, it must not interfere with data transfers between the serial-master and other serial-slave devices. When the serial slave is not selected, its txd output is buffered, resulting in a high impedance drive onto the SPI master rxd line. The buffers shown in the **Figure 19-5** figure are external to the SPI controller. spi_oe_n is the SPI slave output enable signal. †

The serial clock that regulates the data transfer is generated by the serial-master device and input to the SPI slave on `sclk_in`. The slave remains in an idle state until selected by the bus master. When not actively transmitting data, the slave must hold its `txd` line in a high impedance state to avoid interference with serial transfers to other slave devices. The SPI slave output enable (`ss_oe_n`) signal is available for use to control the `txd` output buffer. The slave continues to transfer data to and from the master device as long as it is selected. If the master transmits to all serial slaves, a control bit (`SLV_OE`) in the SPI control register 0 (`CTRLR0`) can be programmed to inform the slave if it should respond with data from its `txd` line. †

Figure 19-5: SPI Slave



The `slv_oe` bit in the control register is only valid if the SPI slave interface is routed to the FPGA. To use the SPI slave in a multi master system or in a system that requires the SPI slave TXD to be tri-stated, you can do the following:

- If you want the SPI slave to control the tri-state of TXD, it must be routed to the FPGA first and use the FPGA IO. HPS I/O can also be used via the Loan I/O interface (timing permitting).
- If you do not want to route to the FPGA, then software control of the TXD (tri-state) must be performed with the already included code to control via an HPS GPIO input. Please refer to the pin connection guidelines to find which GPIO pins correspond to which HPS SPI slave SS ports.

Slave SPI and SSP Serial Transfers

"Motorola SPI Protocol" and the "Texas Instruments Synchronous Serial Protocol (SSP)" contain a description of the SPI and SSP serial protocols, respectively. †

If the SPI slave is *receive only* (TMOD=2), the transmit FIFO buffer need not contain valid data because the data currently in the transmit shift register is resent each time the slave device is selected. The TXE

error flag in the status register (SR) is not set when TMOD=2. You should mask the Transmit FIFO Empty Interrupt when this mode is used. †

If the SPI slave transmits data to the master, you must ensure that data exists in the transmit FIFO buffer before a transfer is initiated by the serial-master device. If the master initiates a transfer to the SPI slave when no data exists in the transmit FIFO buffer, an error flag (TXE) is set in the SPI status register, and the previously transmitted data frame is resent on txd. For continuous data transfers, you must ensure that the transmit FIFO buffer does not become empty before all the data have been transmitted. The transmit FIFO threshold level register (TXFTLR) can be used to early interrupt (Transmit FIFO Empty Interrupt) the processor, indicating that the transmit FIFO buffer is nearly empty. When a DMA Controller is used, the DMA transmit data level register (DMATDLR) can be used to early request the DMA Controller, indicating that the transmit FIFO buffer is nearly empty. The FIFO buffer can then be refilled with data to continue the serial transfer. †

The receive FIFO buffer should be read each time the receive FIFO buffer generates a FIFO full interrupt request to prevent an overflow. The receive FIFO threshold level register (RXFTLR) can be used to give early indication that the receive FIFO buffer is nearly full. When a DMA Controller is used, the DMA receive data level register (DMARDLR) can be used to early request the DMA controller, indicating that the receive FIFO buffer is nearly full. †

Related Information

- **Motorola SPI Protocol** on page 19-16
- **Texas Instruments Synchronous Serial Protocol (SSP)** on page 19-18

Serial Transfers

"National Semiconductor Microwire Protocol" describes the Microwire serial protocol in detail, including timing diagrams and information about how data are structured in the transmit and receive FIFO buffers before and after a serial transfer. The Microwire protocol operates in much the same way as the SPI protocol. There is no decode of the control frame by the SPI slave device. †

Related Information

National Semiconductor Microwire Protocol on page 19-19

Glue Logic for Master Port ss_in_n

When configured as a master, the SPI has an input, `ssi_in_n`, which can be used by the deciding logic in a multi-master system to disable one master when another has priority. The polarity of this signal depends on the serial protocol in use, and the protocol is dynamically selectable.

The table below lists the three protocols and the effect of `ss_in_n` on the ability of the master to transfer data. Note that for the SSP protocol the effect of `ss_in_n` is inverted with respect to the other protocols.

Protocol	ss_in_n value	Effect on Serial Transfer
Motorola SPI	1	Enabled
	0	Disabled
National Semiconductor Microwire	1	Enabled
	0	Disabled
Texas Instruments Serial Protocol (SSP)	1	Disabled
	0	Enabled

Figure 19-6: Arbitration Between Multiple Serial Masters

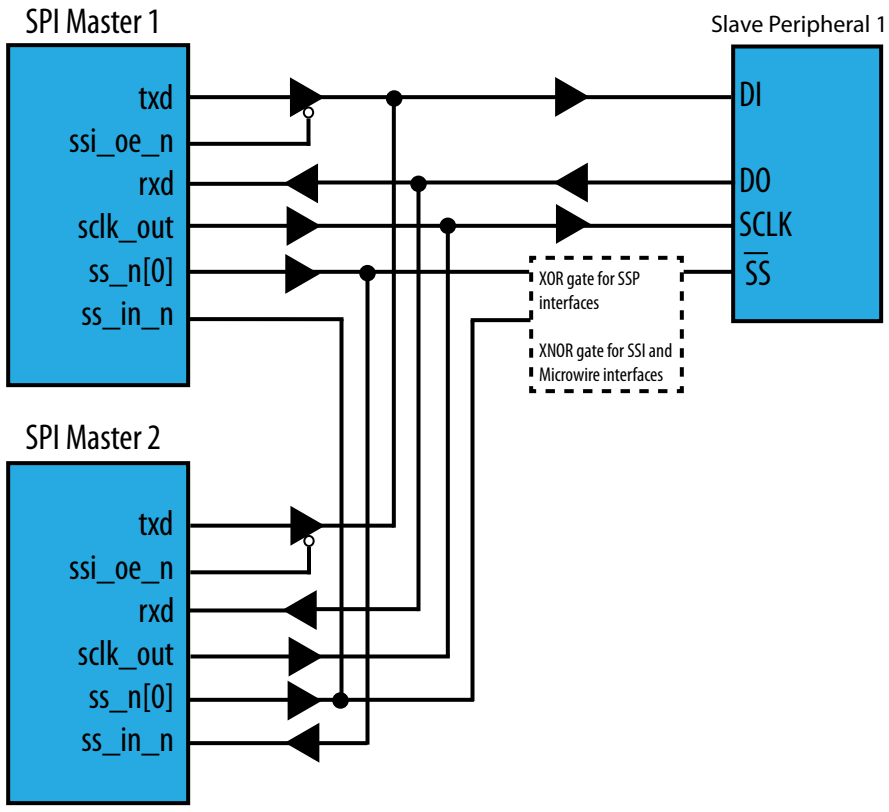
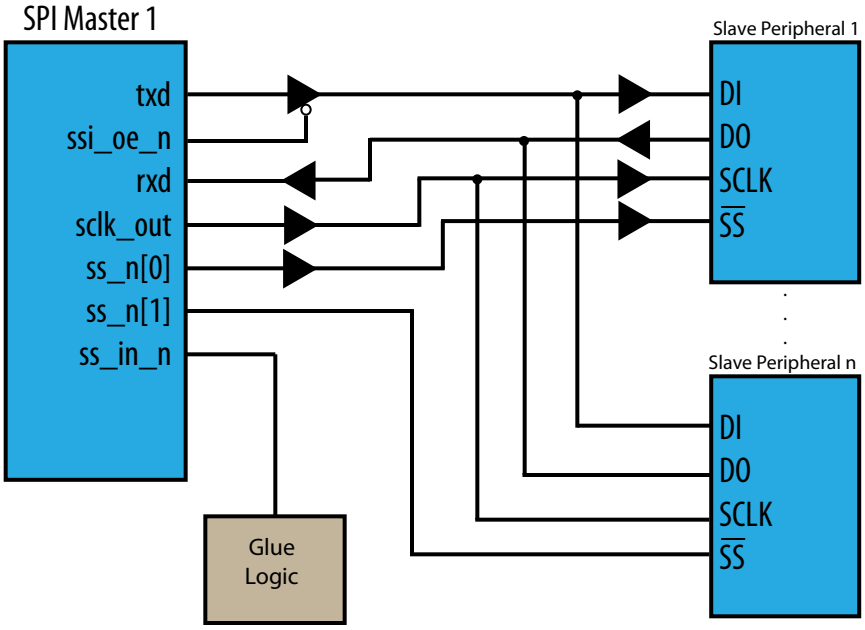


Figure 19-7: SPI Configured as Master Device



Partner Connection Interfaces

The SPI can connect to any serial-master or serial-slave peripheral device using one of several interfaces.

Motorola SPI Protocol

With SPI, the clock polarity (SCPOL) configuration parameter determines whether the inactive state of the serial clock is high or low. The data frame can be 4 to 16 bits in length. †

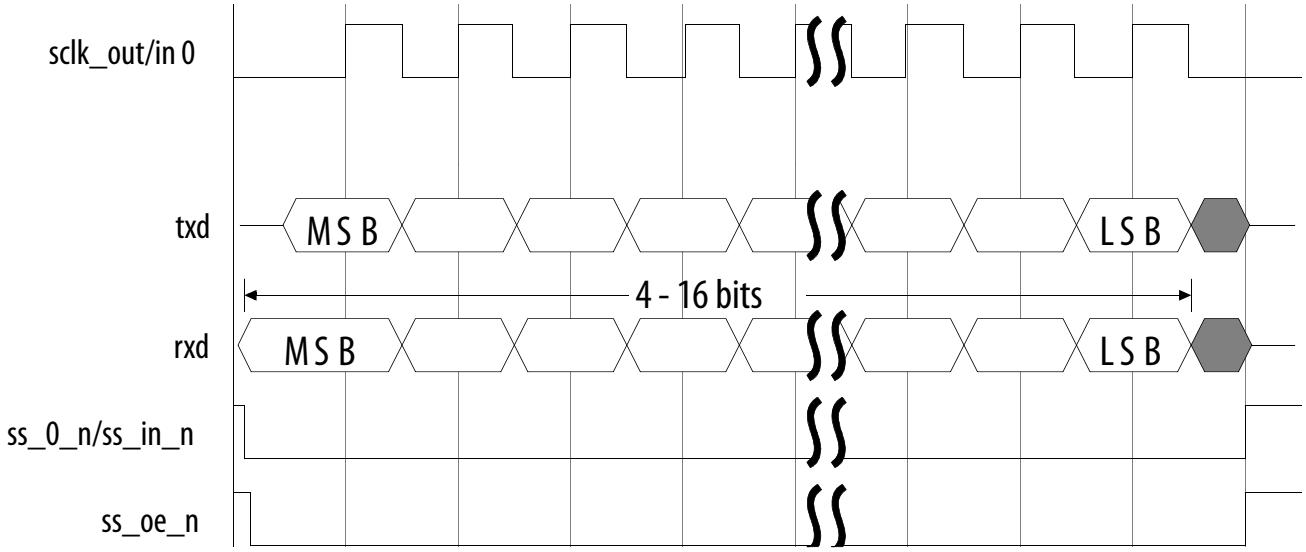
When the configuration parameter (SCPH = 0), data transmission begins on the falling edge of the slave select signal. The first data bit is captured by the master and slave peripherals on the first edge of the serial clock; therefore, valid data must be present on the `txd` and `rxd` lines prior to the first serial clock edge. †

The slave select signal takes effect only when used as slave SPI. For master SPI, the data transmission begins as soon as the output enable signal is deasserted.

The following signals are illustrated in the timing diagrams in this section: †

- sclk_out—serial clock from SPI master †
- sclk_in—serial clock from SPI slave †
- ss_0_n—slave select signal from SPI master †
- ss_oe_n—output enable for the SPI master or slave †
- txd—transmit data line for the SPI master or slave †
- rxd—receive data line for the SPI master or slave †

Figure 19-8: SPI Serial Format (SCPH = 0)



There are four possible transfer modes on the SPI controller for performing SPI serial transactions; refer to "Transfer Modes" . For *transmit and receive transfers* (transfer mode field (9:8) of the Control Register 0 = 0), data transmitted from the SPI controller to the external serial device is written into the transmit FIFO buffer. Data received from the external serial device into the SPI controller is pushed into the receive FIFO buffer. †

Note: For *transmit only* transfers (transfer mode field (9:8) of the Control Register 0 = 1), data transmitted from the SPI controller to the external serial device is written into the transmit FIFO buffer. As the data received from the external serial device is deemed invalid, it is not stored in the SPI receive FIFO buffer. †

For *receive only* transfers (transfer mode field (9:8) of the Control Register 0 = 2), data transmitted from the SPI controller to the external serial device is invalid, so a single dummy word is written into the transmit FIFO buffer to begin the serial transfer. The txd output from the SPI controller is held at a constant logic level for the duration of the serial transfer. Data received from the external serial device into the SPI controller is pushed into the receive FIFO buffer. †

For EEPROM read transfers (transfer mode field [9:8] of the Control Register 0 = 3), opcode and/or EEPROM address are written into the transmit FIFO buffer. During transmission of these control frames, received data is not captured by the SPI master. After the control frames have been transmitted, receive data from the EEPROM is stored in the receive FIFO buffer.

SPI Serial Format

Two different modes of continuous data transfers are supported when SCPH = 0 and 1.†

- When clock phase SCPH = 0, the SPI Controller deasserts the slave select signal between each data word and the serial clock is held to its default value while the slave select signal is deasserted.†
- When SCPH = 1, the slave select is held asserted (active low) for the duration of the transfer.†

Figure 19-9: Serial Format Continuous Transfers (SCPH =0)

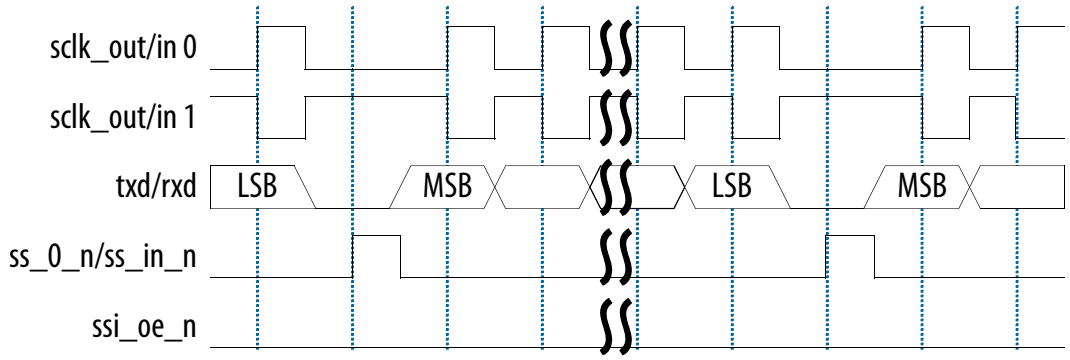
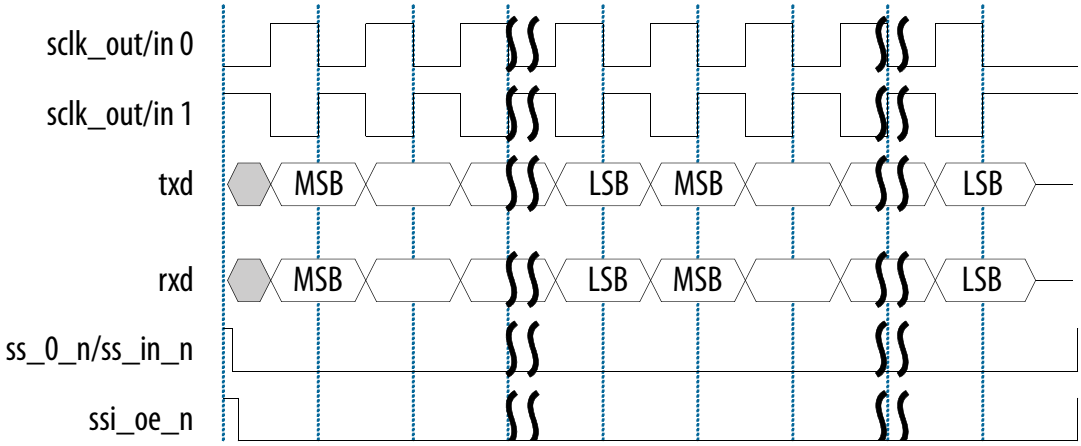


Figure 19-10: Serial Format Continuous Transfers (SCPH =1)



Related Information

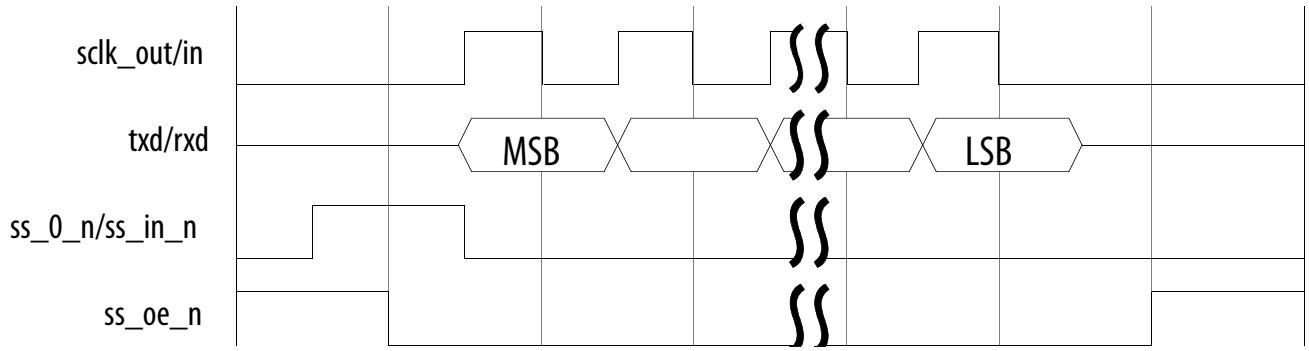
Transfer Modes on page 19-8

Texas Instruments Synchronous Serial Protocol (SSP)

Data transfers begin by asserting the frame indicator line (`ss_0_n`) for one serial clock period. Data to be transmitted are driven onto the `txd` line one serial clock cycle later; similarly data from the slave are driven onto the `rxd` line. Data are propagated on the rising edge of the serial clock (`sclk_out/sclk_in`) and captured on the falling edge. The length of the data frame ranges from 4 to 16 bits.

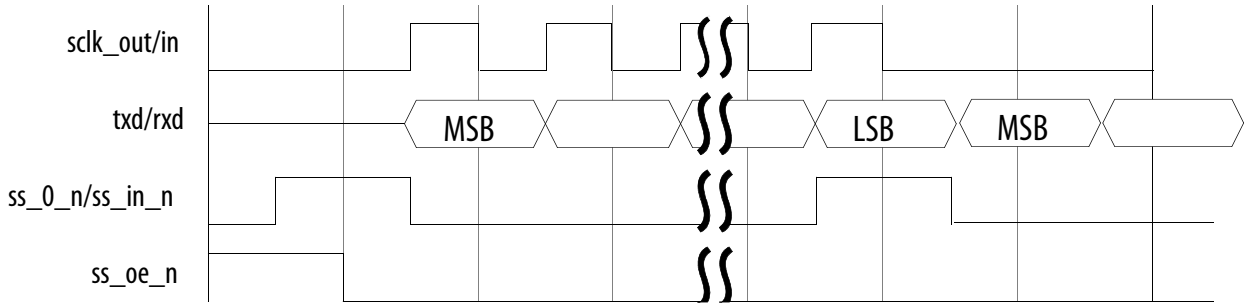
Note: The slave select signal (`ss_0_n`) takes effect only when used as slave SPI. For master SPI, the data transmission begins as soon as the output enable signal is deasserted.

Figure 19-11: SSP Serial Format



Continuous data frames are transferred in the same way as single data frames. The frame indicator is asserted for one clock period during the same cycle as the LSB from the current transfer, indicating that another data frame follows. †

Figure 19-12: SSP Serial Format Continuous Transfer



National Semiconductor Microwire Protocol

For the master SPI, data transmission begins as soon as the output enable signal is deasserted. One-half serial clock (`sclk_out`) period later, the first bit of the control is sent out on the `txd` line. The length of the control word can be in the range 1 to 16 bits and is set by writing bit field CFS (bits 15:12) in `CTRLR0`. The remainder of the control word is transmitted (propagated on the falling edge of `sclk_ou t`) by the SPI serial master. During this transmission, no data are present (high impedance) on the serial master's `rxd` line. †

The direction of the data word is controlled by the `MDD` bit field (bit 1) in the Microwire Control Register (`MWCR`). When `MDD`=0, this indicates that the SPI serial master receives data from the external serial slave. One clock cycle after the LSB of the control word is transmitted, the slave peripheral responds with a dummy 0 bit, followed by the data frame, which can be 4 to 16 bits in length. Data are propagated on the falling edge of the serial clock and captured on the rising edge. †

Continuous transfers from the Microwire protocol can be sequential or nonsequential, and are controlled by the `MWMOD` bit field (bit 0) in the `MWCR`. †

Nonsequential continuous transfers occur, with the control word for the next transfer following immediately after the LSB of the current data word. †

The only modification needed to perform a continuous nonsequential transfer is to write more control words into the transmit FIFO buffer. †

During sequential continuous transfers, only one control word is transmitted from the SPI master. The transfer is started in the same manner as with nonsequential read operations, but the cycle is continued to read further data. The slave device automatically increments its address pointer to the next location and continues to provide data from that location. Any number of locations can be read in this manner; the SPI master terminates the transfer when the number of words received is equal to the value in the CTRLR1 register plus one. †

When MDD = 1, this indicates that the SPI serial master transmits data to the external serial slave. Immediately after the LSB of the control word is transmitted, the SPI master begins transmitting the data frame to the slave peripheral. †

Note: The SPI controller does not support continuous sequential Microwire writes, where MDD = 1 and MWMOD = 1. †

Continuous transfers occur with the control word for the next transfer following immediately after the LSB of the current data word.

The Microwire handshaking interface can also be enabled for SPI master write operations to external serial-slave devices. To enable the handshaking interface, you must write 1 into the MHS bit field (bit 2) on the MWCR register. When MHS is set to 1, the SPI serial master checks for a ready status from the slave device before completing the transfer, or transmitting the next control word for continuous transfers. †

After the first data word has been transmitted to the serial-slave device, the SPI master polls the rxd input waiting for a ready status from the slave device. Upon reception of the ready status, the SPI master begins transmission of the next control word. After transmission of the last data frame has completed, the SPI master transmits a start bit to clear the ready status of the slave device before completing the transfer. †

In the SPI slave, data transmission begins with the falling edge of the slave select signal (ss_in_0). One-half serial clock (sclk_in) period later, the first bit of the control is present on the rxd line. The length of the control word can be in the range of 1 to 16 bits and is set by writing bit field CFS in the CTRLR0 register. The CFS bit field must be set to the size of the expected control word from the serial master. The remainder of the control word is received (captured on the rising edge of sclk_in) by the SPI serial slave. During this reception, no data are driven (high impedance) on the serial slave's txd line. †

The direction of the data word is controlled by the MDD bit field (bit 1) MWCR register. When MDD=0, this indicates that the SPI serial slave is to receive data from the external serial master. Immediately after the control word is transmitted, the serial master begins to drive the data frame onto the SPI slave rxd line. Data are propagated on the falling edge of the serial clock and captured on the rising edge. The slave-select signal is held active-low during the transfer and is deasserted one-half clock cycle later after the data are transferred. The SPI slave output enable signal is held inactive for the duration of the transfer. †

When MDD=1, this indicates that the SPI serial slave transmits data to the external serial master. Immediately after the LSB of the control word is transmitted, the SPI slave transmits a dummy 0 bit, followed by the 4- to 16-bit data frame on the txd line. †

Continuous transfers for a SPI slave occur in the same way as those specified for the SPI master. The SPI slave does not support the handshaking interface, as there is never a busy period. †

Figure 19-13: Single SPI Serial Master Microwire Serial Transfer (MDD=0)

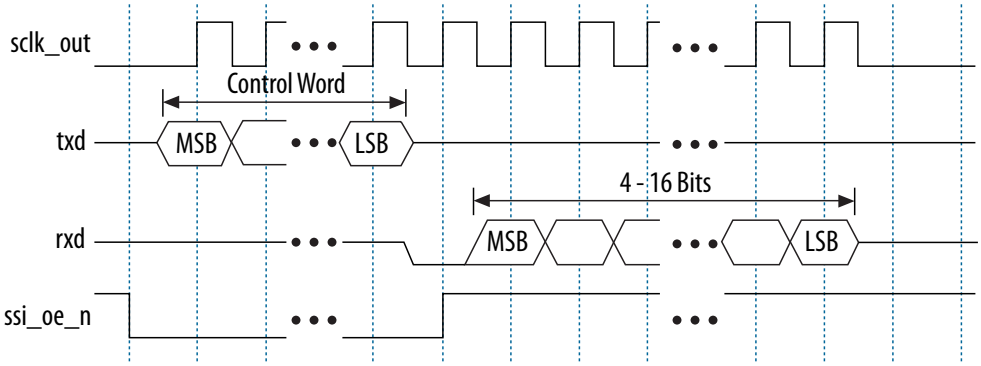
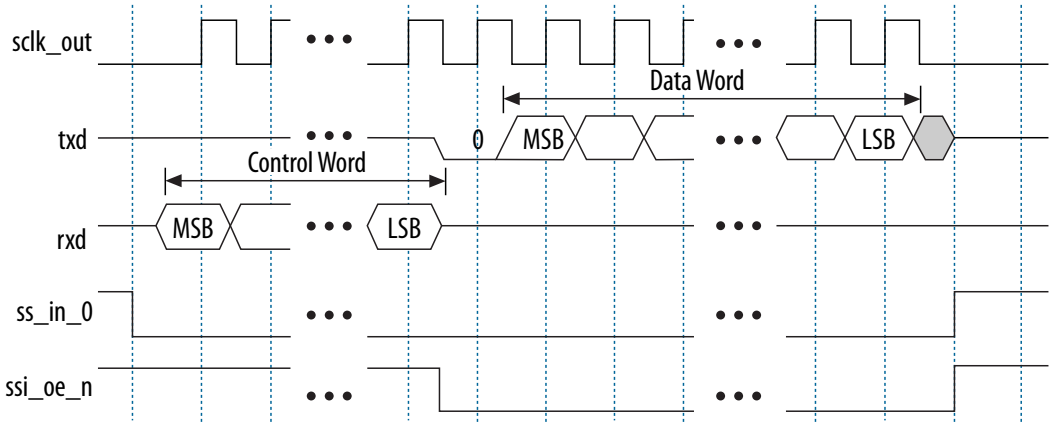


Figure 19-14: Single SPI Slave Microwire Serial Transfer (MDD=1)



DMA Controller Interface

The SPI controller supports DMA signaling to indicate when the receive FIFO buffer has data ready to be read or when the transmit FIFO buffer needs data. It requires two DMA channels, one for transmit data and one for receive data. The SPI controller can issue single or burst DMA transfers and accepts burst acknowledges from the DMA. System software can trigger the DMA burst mode by programming an appropriate value into the threshold registers. The typical setting of the threshold register value is half full.

To enable the DMA Controller interface on the SPI controller, you must write the DMA Control Register (DMACR). Writing a 1 into the TDMAE bit field of DMACR register enables the SPI transmit handshaking interface. Writing a 1 into the RDMAE bit field of the DMACR register enables the SPI receive handshaking interface. †

Slave Interface

The host processor accesses data, control, and status information about the SPI controller through the slave interface. The SPI supports a data bus width of 32 bits.

Control and Status Register Access

Control and status registers within the SPI controller are byte-addressable. The maximum width of the control or status register in the SPI controller is 16 bits. Therefore, all read and write operations to the SPI control and status registers require only one access. †

Data Register Access

The data register (DR) within the SPI controller is 16 bits wide in order to remain consistent with the maximum serial transfer size (data frame). A write operation to DR moves data from the slave write data bus into the transmit FIFO buffer. An read operation from DR moves data from the receive FIFO buffer onto the slave readback data bus. †

Note: The DR register in the SPI controller occupies sixty-four 32-bit locations of the memory map to facilitate burst transfers. There are no burst transactions on the system bus itself, but SPI supports bursts on the system interconnect. Writing to any of these address locations has the same effect as pushing the data from the slave write data bus into the transmit FIFO buffer. Reading from any of these locations has the same effect as popping data from the receive FIFO buffer onto the slave readback data bus. The FIFO buffers on the SPI controller are not addressable.

Clocks and Resets

The SPI controller uses the clock and reset signals shown in the following table.

Table 19-6: SPI Controller Clocks and Resets

	Master	Slave
SPI clock	l4_main_clk	l4_main_clk
SPI bit-rate clock	sclk_out	sclk_in
Reset	spim_rst_n	spis_rst_n

Clock Gating

You can locally clock gate the l4_main_clk to the Master SPI by programming the spimclken bit of the en register in the perpllgroup.

Note: This option is not available for SPI slaves.

Related Information

Clock Manager Address Map and Register Definitions on page 2-18

Taking the SPI Controller Out of Reset

When a cold or warm reset is issued in the HPS, the reset manager resets this module and holds it in reset until software releases it.

After the Cortex-A9 MPCore CPU boots, it can deassert the reset signal by clearing the appropriate bits in the reset manager's corresponding reset register. For details about reset registers, refer to "Module Reset Signals".

Related Information

Modules Requiring Software Deassert on page 3-13

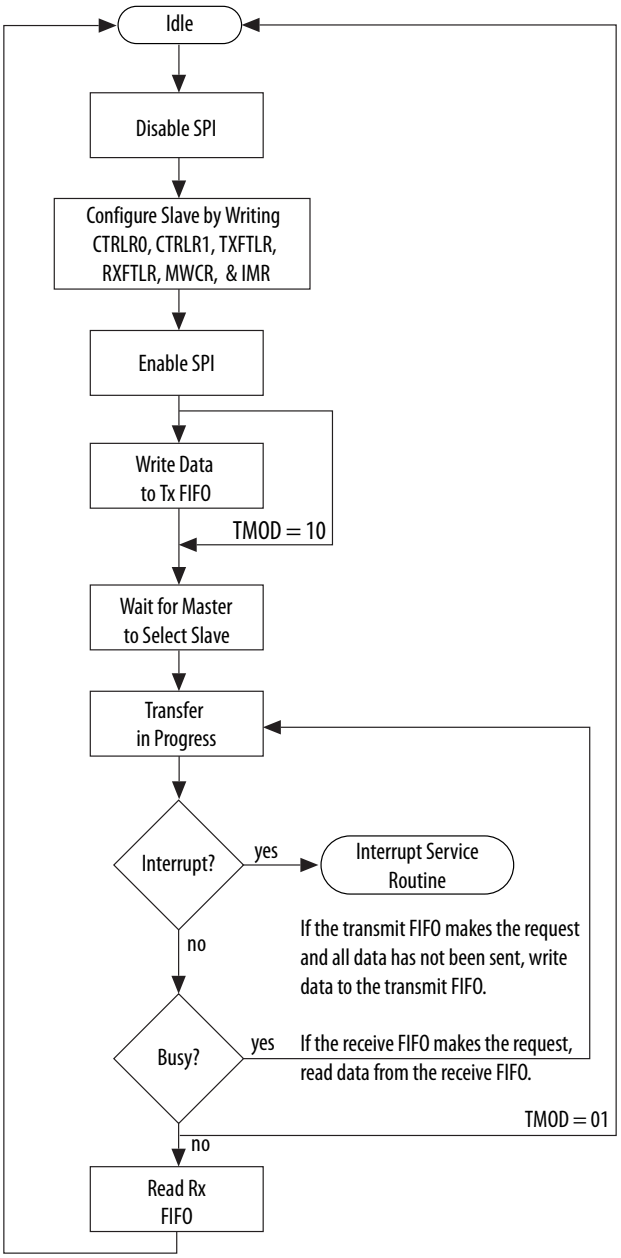
SPI Programming Model

This section describes the programming model for the SPI controller, for the following master and slave transfer types:

- Master SPI and SSP serial transfers
- Master Microwire serial transfers
- Slave SPI and SSP serial transfers
- Slave Microwire serial transfers
- Software Control for slave selection

Master SPI and SSP Serial Transfers

Figure 19-15: Master SPI or SSP Serial Transfer Software Flow



To complete an SPI or SSP serial transfer from the SPI master, follow these steps:

1. If the SPI master is enabled, disable it by writing 0 to the SSI Enable register (`SSIENR`).
2. Set up the SPI master control registers for the transfer. You can set these registers in any order.

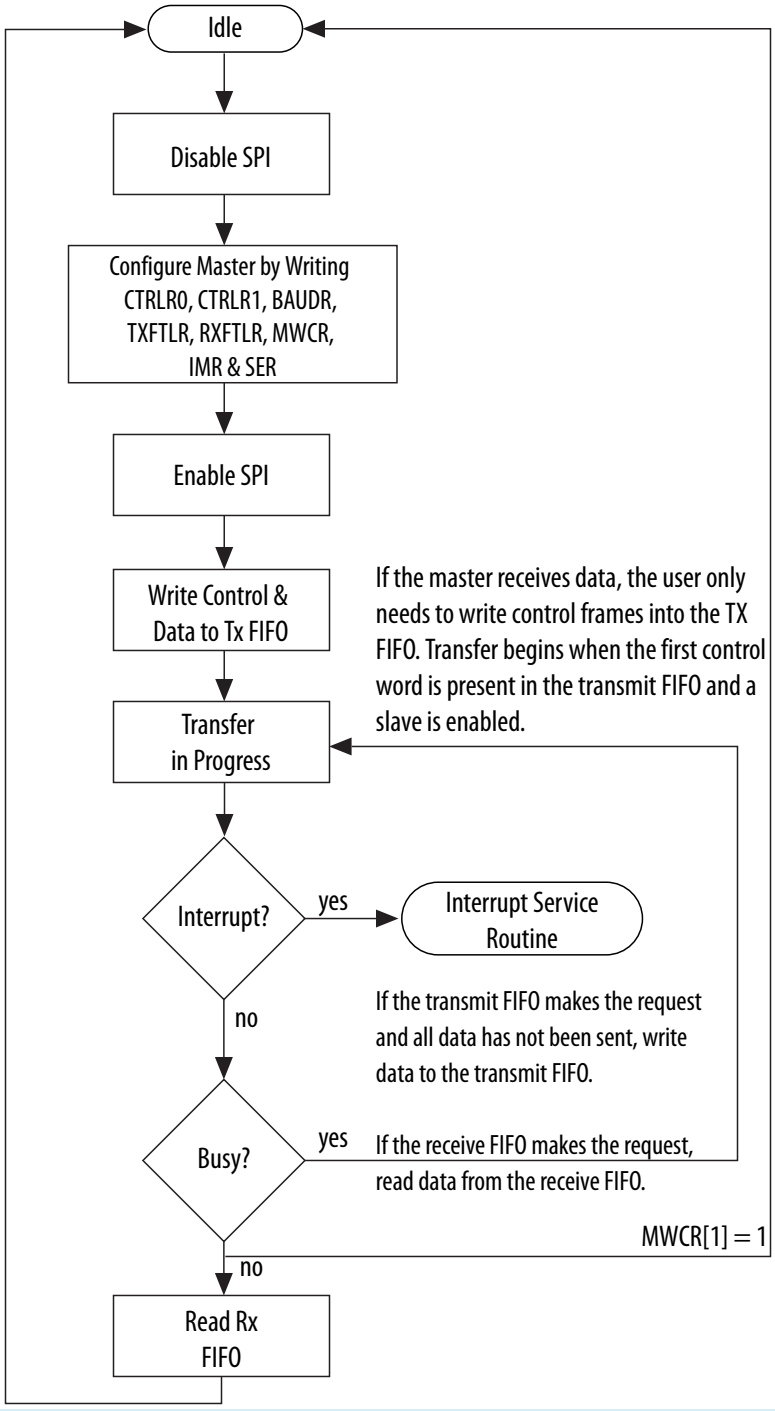
- Write Control Register 0 (CTRLLR0). For SPI transfers, you must set the serial clock polarity and serial clock phase parameters identical to the target slave device.
- If the transfer mode is receive only, write Control Register 1 (CTRLR1) with the number of frames in the transfer minus 1. For example, if you want to receive four data frames, write 3 to this register.
- Write the Baud Rate Select Register (BAUDR) to set the baud rate for the transfer.
- Write the Transmit and Receive FIFO Threshold Level registers (TXFTLR and RXFTLR) to set FIFO buffer threshold levels.
- Write the IMR register to set up interrupt masks.
- Write the Slave Enable Register (SER) register to enable the target slave for selection. If a slave is enabled at this time, the transfer begins as soon as one valid data entry is present in the transmit FIFO buffer. If no slaves are enabled prior to writing to the Data Register (DR), the transfer does not begin until a slave is enabled.

3. Enable the SPI master by writing 1 to the SSIENR register.
4. Write data for transmission to the target slave into the transmit FIFO buffer (write DR). If no slaves were enabled in the SER register at this point, enable it now to begin the transfer.
5. Poll the BUSY status to wait for the transfer to complete. If a transmit FIFO empty interrupt request is made, write the transmit FIFO buffer (write DR). If a receive FIFO full interrupt request is made, read the receive FIFO buffer (read DR).

6. The shift control logic stops the transfer when the transmit FIFO buffer is empty. If the transfer mode is receive only (TMOD = 2), the shift control logic stops the transfer when the specified number of frames have been received. When the transfer is done, the BUSY status is reset to 0.
7. If the transfer mode is not transmit only (TMOD is not equal to 1), read the receive FIFO buffer until it is empty
8. Disable the SPI master by writing 0 to SSIENR.

Master Microwire Serial Transfers

Figure 19-16: Microwire Serial



To complete a Microwire serial transfer from the SPI master, follow these steps:

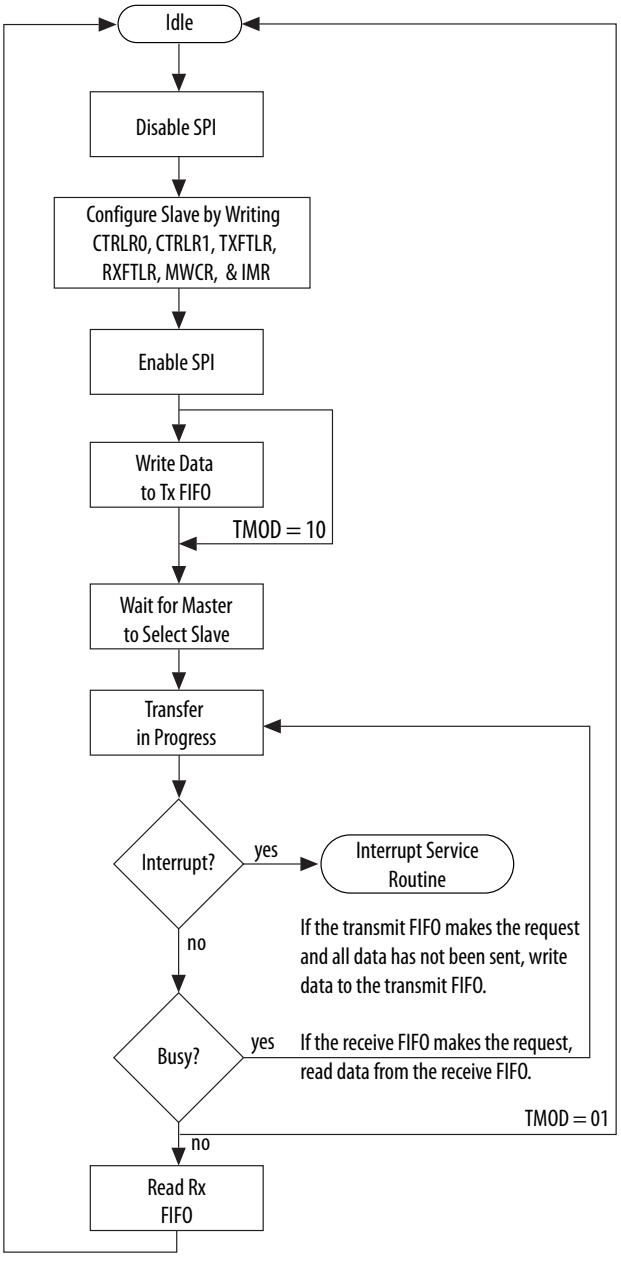
1. If the SPI master is enabled, disable it by writing 0 to SSIENR.
2. Set up the SPI control registers for the transfer. You can set these registers in any order.
   - Write CTRLR0 to set transfer parameters. If the transfer is sequential and the SPI master receives data, write CTRLR1 with the number of frames in the transfer minus 1. For example, if you want to receive four data frames, write 3 to this register.
   - Write BAUDR to set the baud rate for the transfer.
   - Write TXFTLR and RXFTLR to set FIFO buffer threshold levels.
   - Write the IMR register to set up interrupt masks.

   You can write the SER register to enable the target slave for selection. If a slave is enabled here, the transfer begins as soon as one valid data entry is present in the transmit FIFO buffer. If no slaves are enabled prior to writing to the DR register, the transfer does not begin until a slave is enabled.
3. Enable the SPI master by writing 1 to the SSIENR register.
4. If the SPI master transmits data, write the control and data words into the transmit FIFO buffer (write DR). If the SPI master receives data, write the control word or words into the transmit FIFO buffer. If no slaves were enabled in the SER register at this point, enable now to begin the transfer.
5. Poll the BUSY status to wait for the transfer to complete. If a transmit FIFO empty interrupt request is made, write the transmit FIFO buffer (write DR). If a receive FIFO full interrupt request is made, read the receive FIFO buffer (read DR).

6. The shift control logic stops the transfer when the transmit FIFO buffer is empty. If the transfer mode is sequential and the SPI master receives data, the shift control logic stops the transfer when the specified number of data frames is received. When the transfer is done, the BUSY status is reset to 0.
7. If the SPI master receives data, read the receive FIFO buffer until it is empty.
8. Disable the SPI master by writing 0 to SSIENR.

Slave SPI and SSP Serial Transfers

Figure 19-17: Slave SPI or SSP Serial Transfer Software Flow



To complete a continuous serial transfer from a serial master to the SPI slave, follow these steps:

1. If the SPI slave is enabled, disable it by writing 0 to SSIENR.
2. Set up the SPI control registers for the transfer. You can set these registers in any order.
   - Write CTRLR0 (for SPI transfers, set SCPH and SCPOL identical to the master device).
   - Write TXFTLR and RXFTLR to set FIFO buffer threshold levels.
   - Write the IMR register to set up interrupt masks.
3. Enable the SPI slave by writing 1 to the SSIENR register.
4. If the transfer mode is transmit and receive (TMOD= 0) or transmit only (TMOD= 1), write data for transmission to the master into the transmit FIFO buffer (write DR). If the transfer mode is receive only (TMOD= 2), you need not write data into the transmit FIFO buffer. The current value in the transmit shift register is retransmitted.
5. The SPI slave is now ready for the serial transfer. The transfer begins when a serial-master device selects the SPI slave.
6. When the transfer is underway, the BUSY status can be polled to return the transfer status. If a transmit FIFO empty interrupt request is made, write the transmit FIFO buffer (write DR). If a receive FIFO full interrupt request is made, read the receive FIFO buffer (read DR).
7. The transfer ends when the serial master removes the select input to the SPI slave. When the transfer is completed, the BUSY status is reset to 0.
8. If the transfer mode is not transmit only (TMOD != 1), read the receive FIFO buffer until empty.
9. Disable the SPI slave by writing 0 to SSIENR.

Slave Microwire Serial Transfers

For the SPI slave, the Microwire protocol operates in much the same way as the SPI protocol. The SPI slave does not decode the control frame.

Software Control for Slave Selection

When using software to select slave devices, the input select lines from serial slave devices is connected to a single slave select output on the SPI master.

Example: Slave Selection Software Flow for SPI Master

1. If the SPI master is enabled, disable it by writing 0 to SSIENR.
2. Write CTRLR0 to match the required transfer.
3. If the transfer is receive only, write the number of frames into CTRLR1.
4. Write BAUDR to set the transfer baud rate.
5. Write TXFTLR and RXFTLR to set FIFO buffer threshold levels.
6. Write IMR register to set interrupt masks.
7. Write SER register bit 0 to 1 to select slave 1 in this example.
8. Write SSIENR register bit 0 to 1 to enable SPI master.

Example: Slave Selection Software Flow for SPI Slave

1. If the SPI slave is enabled, disable it by writing 0 to SSIENR.
2. Write CTRLR0 to match the required transfer.
3. Write TXFTLR and RXFTLR to set FIFO buffer threshold levels.
4. Write IMR register to set interrupt masks.
5. Write SSIENR register bit 0 to 1 to enable SPI slave.
6. If the SPI slave transmits data, write data into TX FIFO buffer.

Note: All other SPI slaves are disabled (SSIENR = 0) and therefore will not respond to an active level on their ss_in_n port.

The FIFO buffer depth (FIFO_DEPTH) for both the RX and TX buffers in the SPI controller is 256 entries.

DMA Controller Operation

To enable the DMA controller interface on the SPI controller, you must write the DMA Control Register (DMACR). Writing a 1 to the TDMAE bit field of DMACR register enables the SPI controller transmit handshaking interface. Writing a 1 to the RDMAE bit field of the DMACR register enables the SPI controller receive handshaking.†

Related Information

DMA Controller on page 16-1
For details about the DMA controller, refer to the *DMA Controller* chapter.

Transmit FIFO Buffer Underflow

During SPI serial transfers, transmit FIFO buffer requests are made to the DMA Controller whenever the number of entries in the transmit FIFO buffer is less or equal to the value in DMA Transmit Data Level Register (DMATDLR); also known as the watermark level. The DMA Controller responds by writing a burst of data to the transmit FIFO buffer, of length specified as DMA burst length.†

Note: Data should be fetched from the DMA often enough for the transmit FIFO buffer to perform serial transfers continuously, that is, when the FIFO buffer begins to empty, another DMA request should be triggered. Otherwise, the FIFO buffer will run out of data (underflow). To prevent this condition, you must set the watermark level correctly.†

Related Information

DMA Controller on page 16-1
For details about the DMA burst length microcode setup, refer to the *DMA Controller* chapter.

Transmit FIFO Watermark

Therefore, the goal in choosing a watermark level is to minimize the number of transactions per block, while at the same time keeping the probability of an underflow condition to an acceptable level. In practice, this is a function of the ratio of the rate at which the SPI transmits data to the rate at which the DMA can respond to destination burst requests. †

Example 1: Transmit FIFO Watermark Level = 64

Consider the example where the assumption is made: †

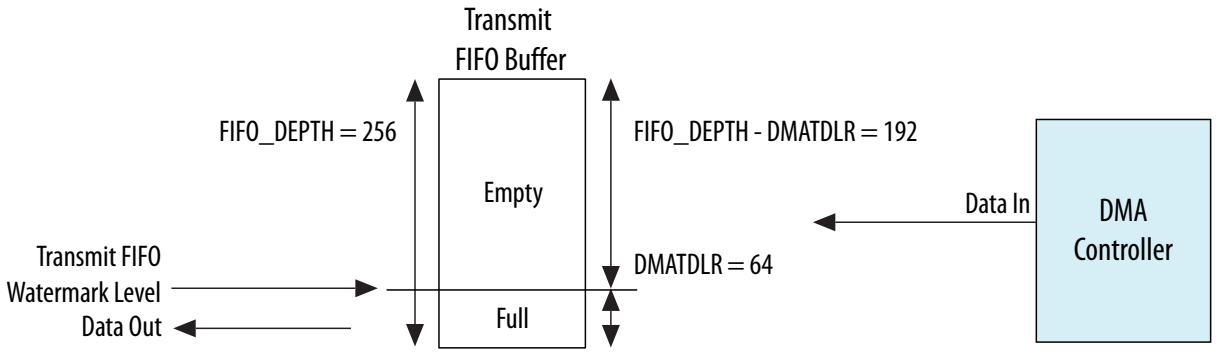
DMA burst length = FIFO_DEPTH - DMATDLR

Here the number of data items to be transferred in a DMA burst is equal to the empty space in the transmit FIFO buffer.

Consider the following:

- Transmit FIFO watermark level = DMATDLR = 64 †
- DMA burst length = FIFO_DEPTH - DMATDLR = 192 †
- SPI transmit FIFO_DEPTH = 256 †
- Block transaction size = 960 †

Figure 19-18: Transmit FIFO Watermark Level = 64



The number of burst transactions needed equals the block size divided by the number of data items per burst:

Block transaction size/DMA burst length = 960/192 = 5

The number of burst transactions in the DMA block transfer is 5. But the watermark level, `DMATDLR`, is quite low. Therefore, there is a high probability that the SPI serial transmit line will need to transmit data when there is no data left in the transmit FIFO buffer. This is a transmit underflow condition. This occurs because the DMA has not had time to service the DMA request before the FIFO buffer becomes empty.

Example 2: Transmit FIFO Watermark Level = 192

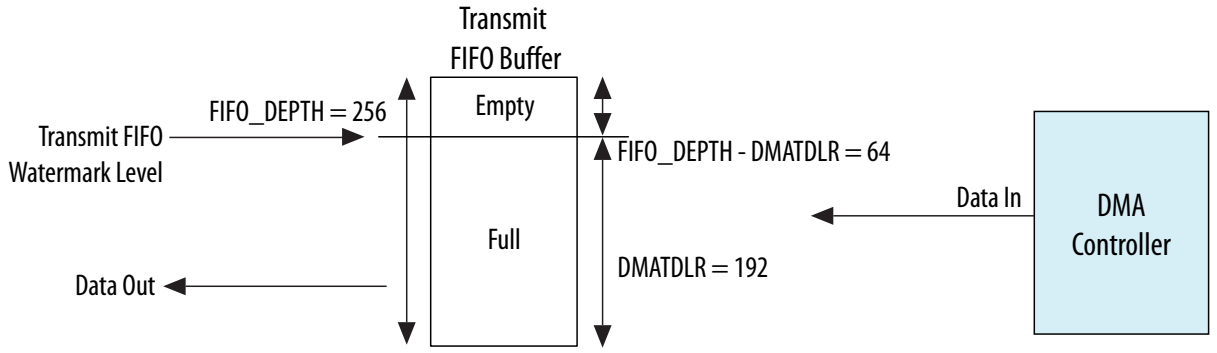
Consider the example where the assumption is made: †

DMA burst length = `FIFO_DEPTH - DMATDLR`

Here the number of data items to be transferred in a DMA burst is equal to the empty space in the transmit FIFO buffer. Consider the following:

- Transmit FIFO watermark level = `DMATDLR` = 192 †
- DMA burst length = `FIFO_DEPTH - DMATDLR` = 64 †
- SPI transmit `FIFO_DEPTH` = 256 †
- Block transaction size = 960 †

Figure 19-19: Transmit FIFO Watermark Level = 192



Number of burst transactions in block: †

Block transaction size/DMA burst length = 960/64 = 15 †

In this block transfer, there are 15 destination burst transactions in a DMA block transfer. But the watermark level, DMATDLR, is high. Therefore, the probability of SPI transmit underflow is low because the DMA controller has plenty of time to service the destination burst transaction request before the SPI transmit FIFO buffer becomes empty. †

This case has a lower probability of underflow at the expense of more burst transactions per block. This provides a potentially greater amount of bursts per block and worse bus utilization than the former case.

Transmit FIFO Buffer Overflow

Setting the DMA transaction burst length to a value greater than the watermark level that triggers the DMA request may cause overflow when there is not enough space in the transmit FIFO buffer to service the destination burst request. Therefore, the following equation must be adhered to in order to avoid overflow: †

DMA burst length <= FIFO_DEPTH - DMATDLR

In Example 2: Transmit Watermark Level = 192, the amount of space in the transmit FIFO buffer at the time of the burst request is made is equal to the DMA burst length. Thus, the transmit FIFO buffer may be full, but not overflowed, at the completion of the burst transaction.

Therefore, for optimal operation, DMA burst length should be set at the FIFO buffer level that triggers a transmit DMA request; that is: †

DMA burst length = FIFO_DEPTH - DMATDLR

Adhering to this equation reduces the number of DMA bursts needed for block transfer, and this in turn improves bus utilization. †

The transmit FIFO buffer will not be full at the end of a DMA burst transfer if the SPI controller has successfully transmitted one data item or more on the serial transmit line during the transfer. †

Related Information

Transmit FIFO Watermark on page 19-30

Receive FIFO Buffer Overflow

During SPI serial transfers, receive FIFO buffer requests are made to the DMA whenever the number of entries in the receive FIFO buffer is at or above the DMA Receive Data Level Register, that is DMATDLR + 1.This is known as the watermark level. The DMA responds by fetching a burst of data from the receive FIFO buffer. †

Data should be fetched by the DMA often enough for the receive FIFO buffer to accept serial transfers continuously, that is, when the FIFO buffer begins to fill, another DMA transfer is requested. Otherwise the FIFO buffer will fill with data (overflow). To prevent this condition, the user must set the watermark level correctly. †

Choosing Receive Watermark Level

Similar to choosing the transmit watermark level, the receive watermark level, DMATDLR + 1, should be set to minimize the probability of overflow. It is a trade off between the number of DMA burst transactions required per block versus the probability of an overflow occurring. †

Related Information

Texas Instruments Synchronous Serial Protocol (SSP) on page 19-18

Receive FIFO Buffer Underflow

Setting the source transaction burst length greater than the watermark level can cause underflow where there is not enough data to service the source burst request. Therefore, the following equation must be adhered to avoid underflow: †

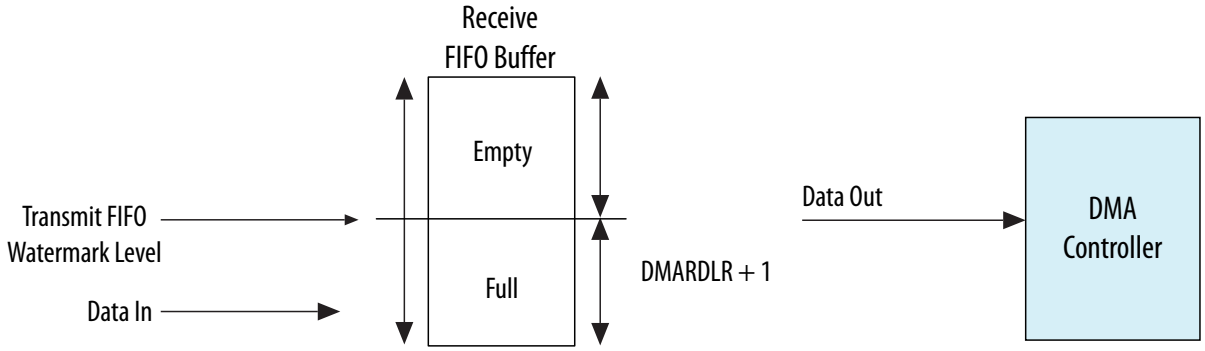
DMA burst length = `DMATDLR` + 1

If the number of data items in the receive FIFO buffer is equal to the source burst length at the time of the burst request is made, the receive FIFO buffer may be emptied, but not underflowed, at the completion of the burst transaction. For optimal operation, DMA burst length should be set at the watermark level, `DMATDLR` + 1. †

Adhering to this equation reduces the number of DMA bursts in a block transfer, which in turn can improve bus utilization. †

Note: The receive FIFO buffer will not be empty at the end of the source burst transaction if the SPI controller has successfully received one data item or more on the serial receive line during the burst. †

Figure 19-20: Receive FIFO Buffer



SPI Controller Address Map and Register Definitions

For complete HPS address map and register definitions, refer to the **Arria 10 HPS Address Map and Register Definitions**.

spim Address Map

Module Instance	Base Address	End Address
i_spim_0_spim	0xFFDA4000	0xFFDA4FFF
i_spim_1_spim	0xFFDA5000	0xFFDFFFFF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
ctrlr0 on page 19-50	0x0	32	RW	0x7	Control Register 0: This register controls the serial data transfer. It is impossible to write to this register when the DW_apb_ssi is enabled. The DW_apb_ssi is enabled and disabled by writing to the SSIENR register.
ctrlr1 on page 19-55	0x4	32	RW	0x0	Control Register 1 This register exists only when the DW_apb_ssi is configured as a master device. When the DW_apb_ssi is configured as a serial slave, writing to this location has no effect; reading from this location returns 0. Control register 1 controls the end of serial transfers when in receive-only mode. It is impossible to write to this register when the DW_apb_ssi is enabled. The DW_apb_ssi is enabled and disabled by writing to the SSIENR register.
spienr on page 19-56	0x8	32	RW	0x0	SSI Enable Register

Register	Offset	Width	Access	Reset Value	Description
mwcr on page 19-57	0xC	32	RW	0x0	Microwire Control Register. This register controls the direction of the data word for the half-duplex Microwire serial protocol. It is impossible to write to this register when the DW_apb_ssi is enabled. The DW_apb_ssi is enabled and disabled by writing to the SSIENR register.
ser on page 19-60	0x10	32	RW	0x0	Slave Enable Register. This register is valid only when the DW_apb_ssi is configured as a master device. When the DW_apb_ssi is configured as a serial slave, writing to this location has no effect; reading from this location returns 0. The register enables the individual slave select output lines from the DW_apb_ssi master. Up to 16 slave-select output pins are available on the DW_apb_ssi master. You cannot write to this register when DW_apb_ssi is busy and when SSI_EN = 1.

Register	Offset	Width	Access	Reset Value	Description
baudr on page 19-61	0x14	32	RW	0x0	Baud Rate Select. This register is valid only when the DW_apb_ssi is configured as a master device. When the DW_apb_ssi is configured as a serial slave, writing to this location has no effect; reading from this location returns 0. The register derives the frequency of the serial clock that regulates the data transfer. The 16-bit field in this register defines the ssi_clk divider value. It is impossible to write to this register when the DW_apb_ssi is enabled. The DW_apb_ssi is enabled and disabled by writing to the SSIENR register.
txftlr on page 19-63	0x18	32	RW	0x0	Transmit FIFO Threshold Level. This register controls the threshold value for the transmit FIFO memory. The DW_apb_ssi is enabled and disabled by writing to the SSIENR register.

Register	Offset	Width	Access	Reset Value	Description
rxftlr on page 19-64	0x1C	32	RW	0x0	Receive FIFO Threshold level. This register controls the threshold value for the receive FIFO memory. The DW_apb_ssi is enabled and disabled by writing to the SSIENR register.
txflr on page 19-65	0x20	32	RO	0x0	Transmit FIFO Level Register
rxflr on page 19-66	0x24	32	RO	0x0	Receive FIFO Level Register
sr on page 19-67	0x28	32	RO	0x6	Status Register
imr on page 19-69	0x2C	32	RW	0x3F	Interrupt Mask Register
isr on page 19-71	0x30	32	RO	0x0	Interrupt Status Register
risr on page 19-73	0x34	32	RO	0x0	Raw Interrupt StatusRegister
txoicr on page 19-76	0x38	32	RO	0x0	Transmit FIFO Overflow Interrupt Clear Register
rxoicr on page 19-76	0x3C	32	RO	0x0	Receive FIFO Overflow Interrupt Clear Register
rxuicr on page 19-77	0x40	32	RO	0x0	Receive FIFO Underflow Interrupt Clear Register
icr on page 19-78	0x48	32	RO	0x0	Interrupt Clear Register

Register	Offset	Width	Access	Reset Value	Description
dmacr on page 19-79	0x4C	32	RW	0x0	DMA Control Register. This register is only valid when DW_apb_ssi is configured with a set of DMA Controller interface signals (SSI_HAS_DMA = 1). When DW_apb_ssi is not configured for DMA operation, this register will not exist and writing to the register's address will have no effect; reading from this register address will return zero. The register is used to enable the DMA Controller interface operation.
dmatdlr on page 19-80	0x50	32	RW	0x0	DMA Transmit Data Level. This register is only valid when the DW_apb_ssi is configured with a set of DMA interface signals (SSI_HAS_DMA = 1). When DW_apb_ssi is not configured for DMA operation, this register will not exist and writing to its address will have no effect; reading from its address will return zero.

Register	Offset	Width	Access	Reset Value	Description
dmardlr on page 19-81	0x54	32	RW	0x0	DMA Receive Data Level. This register is only valid when DW_apb_ssi is configured with a set of DMA interface signals (SSI_HAS_DMA = 1). When DW_apb_ssi is not configured for DMA operation, this register will not exist and writing to its address will have no effect; reading from its address will return zero.
idr on page 19-82	0x58	32	RO	0x5510000	Identification Register. This register contains the peripherals identification code, which is written into the register at configura-tion time using coreConsultant.
spi_version_id on page 19-83	0x5C	32	RW	0x3332322A	coreKit Version ID Register
dr on page 19-84	0x60	32	RW	0x0	The DW_apb_ssi data register is a 16-bit read/write buffer for the transmit/receive FIFOs. When the register is read, data in the receive FIFO buffer is accessed. When it is written to, data are moved into the transmit FIFO buffer; a write can occur only when SSI_EN = 1. FIFOs are reset when SSI_EN = 0.

Register	Offset	Width	Access	Reset Value	Description
rx_sample_dly on page 19-85	0xF0	32	RW	0x0	RX Sample Delay. This register is only valid when the DW_apb_ssi is configured with rxd sample delay logic (SSI_HAS_RX_SAMPLE_DELAY==1). When the DW_apb_ssi is not configured with rxd sample delay logic, this register will not exist and writing to its address location will have no effect; reading from its address will return zero. This register control the number of ssi_clk cycles that are delayed (from the default sample time) before the actual sample of the rxd input occurs. It is impossible to write to this register when the DW_apb_ssi is enabled. The DW_apb_ssi is enabled and disabled by writing to the SSIENR register.
msticr on page 19-86	0x44	32	RO	0x0	Multi-Master Interrupt Clear Register
rsvd_0 on page 19-87	0xF4	32	RO	0x0	RSVD_0 - Reserved address location
rsvd_1 on page 19-87	0xF8	32	RO	0x0	RSVD_1 - Reserved address location
rsvd_2 on page 19-88	0xFC	32	RO	0x0	RSVD_2 - Reserved address location

spim Summary

Module Instance	Base Address
i_spim_0_spim	0xFFDA4000
i_spim_1_spim	0xFFDA5000

Register Address Offset	Bit Fields															
i_spim_0_spim																
ctrlr0 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cfs RW 0x0				srl RW 0x0	Reserved	tmod RW 0x0		scpol RW 0x0	scph RW 0x0	frf RW 0x0		dfs RW 0x7			
ctrlr1 0x4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ndf RW 0x0															
spienr 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															spi_en RW 0x0
mwcr 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													mhs RW 0x0	mdd RW 0x0	mwmod RW 0x0
ser 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved												ser RW 0x0			

Register Address Offset	Bit Fields															
baudr 0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	sckdv RW 0x0															
txftlr 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								tft RW 0x0							
rxftlr 0x1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								rft RW 0x0							
txflr 0x20	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							txtfl RO 0x0								
rxflr 0x24	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							rxtfl RO 0x0								
sr 0x28	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved									dcol RO 0x0	Reserved	rff RO 0x0	rfne RO 0x0	tfe RO 0x1	tfnf RO 0x1	busy RO 0x0

Register / Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
imr 0x2C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved										mstim RW 0x1	rxfim RW 0x1	rxoim RW 0x1	rxuim RW 0x1	txoim RW 0x1	txeim RW 0x1
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
isr 0x30	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved										mstis RO 0x0	rxfis RO 0x0	rxois RO 0x0	rxuis RO 0x0	txois RO 0x0	txeis RO 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
risr 0x34	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved										mstir RO 0x0	rxfir RO 0x0	rxoir RO 0x0	rxuir RO 0x0	txoir RO 0x0	txeir RO 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
txoicr 0x38	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															txoicr RO 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
rxoicr 0x3C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															rxoicr RO 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
rxuicr 0x40	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															rxuicr RO 0x0

Register Address Offset	Bit Fields															
icr 0x48	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															icr RO 0x0
dmacr 0x4C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														tdmae RW 0x0	rdmae RW 0x0
dmatdlr 0x50	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								dmatdl RW 0x0							
dmardlr 0x54	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								dmardl RW 0x0							
idr 0x58	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	idr RO 0x5510000															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	idr RO 0x5510000															
spi_version_id 0x5C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	spi_version_id RW 0x3332322A															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	spi_version_id RW 0x3332322A															

Register Address Offset	Bit Fields															
dr 0x60	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	dr RW 0x0															
rx_sample_dly 0xF0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								rsd RW 0x0							
msticr 0x44	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															msticr RO 0x0
rsvd_0 0xF4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															
rsvd_1 0xF8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															
rsvd_2 0xFC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															
i_spim_1_spim																
ctrlr0 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cfs RW 0x0				srl RW 0x0	Reserved	tmod RW 0x0		scpol RW 0x0	scph RW 0x0	frf RW 0x0		dfs RW 0x7			

Register Address Offset	Bit Fields															
ctrlr1 0x4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ndf RW 0x0															
spienr 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															spi_en RW 0x0
mwcr 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													mhs RW 0x0	mdd RW 0x0	mwmod RW 0x0
ser 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved												ser RW 0x0			
baudr 0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	sckdv RW 0x0															
txftlr 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								tft RW 0x0							
rxftlr 0x1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								rft RW 0x0							

Register Address Offset	Bit Fields															
txflr 0x20	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							txtfl RO 0x0								
rxflr 0x24	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							rxtfl RO 0x0								
sr 0x28	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved									dcol RO 0x0	Reserved	rff RO 0x0	rfne RO 0x0	tfe RO 0x1	tfnf RO 0x1	busy RO 0x0
imr 0x2C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved										mstim RW 0x1	rxfim RW 0x1	rxoim RW 0x1	rxuim RW 0x1	txoim RW 0x1	txeim RW 0x1
isr 0x30	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved										mstis RO 0x0	rxfis RO 0x0	rxois RO 0x0	rxuis RO 0x0	txois RO 0x0	txeis RO 0x0
risr 0x34	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved										mstir RO 0x0	rxfir RO 0x0	rxoir RO 0x0	rxuir RO 0x0	txoir RO 0x0	txeir RO 0x0

Register Address Offset	Bit Fields															
txoicr 0x38	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															txoicr RO 0x0
rxoicr 0x3C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															rxoicr RO 0x0
rxuicr 0x40	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															rxuicr RO 0x0
icr 0x48	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															icr RO 0x0
dmacr 0x4C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														tdmae RW 0x0	rdmae RW 0x0
dmatdlr 0x50	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								dmatdl RW 0x0							
dmardlr 0x54	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								dmardl RW 0x0							

Register Address Offset	Bit Fields															
idr 0x58	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	idr RO 0x5510000															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	idr RO 0x5510000															
spi_version_id 0x5C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	spi_version_id RW 0x3332322A															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	spi_version_id RW 0x3332322A															
dr 0x60	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	dr RW 0x0															
rx_sample_dly 0xF0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								rsd RW 0x0							
msticr 0x44	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															msticr RO 0x0
rsvd_0 0xF4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															
rsvd_1 0xF8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
rsvd_2	Reserved															
0xFC	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															

ctrlr0

Control Register 0:
This register controls the serial data transfer. It is impossible to write to this register when the DW_apb_ssi is enabled. The DW_apb_ssi is enabled and disabled by writing to the SSIENR register.

Module Instance	Base Address	Register Address
i_spim_0_spim	0xFFDA4000	0xFFDA4000
i_spim_1_spim	0xFFDA5000	0xFFDA5000

Offset: 0x0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cfs RW 0x0				srl RW 0x0	Reserved	tmod RW 0x0		scpol RW 0x0	scph RW 0x0	frf RW 0x0		dfs RW 0x7			

ctrlr0 Fields

Bit	Name	Description	Access	Reset
15:12	cfs	Control Frame Size. Selects the length of the control word for the Microwire frame format Value: Description 0xa: SIZE11BIT 0xb: SIZE12BIT 0xc: SIZE13BIT 0xd: SIZE14BIT 0xe: SIZE15BIT 0xf: SIZE16BIT 0x0: SIZE1BIT 0x1: SIZE2BIT 0x2: SIZE3BIT 0x3: SIZE4BIT 0x4: SIZE5BIT 0x05: SIZE6BIT 0x6: SIZE7BIT 0x7: SIZE8BIT 0x8: SIZE9BIT 0x9: SIZE10BIT	RW	0x0
11	srl	Shift Register Loop. Used for testing purposes only. When internally active, connects the transmit shift register output to the receive shift register input. 0 - Normal Mode Operation 1 - Test Mode Operation Value: Description 0x0: NORMMODE 0x1: TESTMODE	RW	0x0

Bit	Name	Description	Access	Reset
9:8	tmod	Transfer Mode. Selects the mode of transfer for serial communication. This field does not affect the transfer duplicity. Only indicates whether the receive or transmit data are valid. In transmit-only mode, data received from the external device is not valid and is not stored in the receive FIFO memory; it is overwritten on the next transfer. In receive-only mode, transmitted data are not valid. After the first write to the transmit FIFO, the same word is retransmitted for the duration of the transfer. In transmit-and-receive mode, both transmit and receive data are valid. The transfer continues until the transmit FIFO is empty. Data received from the external device are stored into the receive FIFO memory, where it can be accessed by the host processor. 00 - Transmit & Receive 01 - Transmit Only 10 - Receive Only 11 - Reserved **Value** **Description** 0x0 TXRX 0x1 TXONLY 0x2 RXONLY 0x3 EERD	RW	0x0

Bit	Name	Description	Access	Reset
7	scpol	Serial Clock Polarity. Valid when the frame format (FRF) is set to Motorola SPI. Used to select the polarity of the inactive serial clock, which is held inactive when the DW_apb_ssi master is not actively transferring data on the serial bus. 0 - Inactive state of serial clock is low 1 - Inactive state of serial clock is high **Value** **Description** 0x0 INACTIVELOW 0x1 INACTIVEHIGH	RW	0x0
6	scph	Serial Clock Phase. Valid when the frame format (FRF) is set to Motorola SPI. The serial clock phase selects the relation-ship of the serial clock with the slave select signal. When SCPH = 0, data are captured on the first edge of the serial clock. When SCPH = 1, the serial clock starts toggling one cycle after the slave select line is activated, and data are captured on the second edge of the serial clock. 0: Serial clock toggles in middle of first data bit 1: Serial clock toggles at start of first data bit **Value** **Description** 0x0 MIDBIT 0x1 STARTBIT	RW	0x0

Bit	Name	Description	Access	Reset
5:4	frf	Frame Format. Selects which serial protocol transfers the data. 00 - Motorola SPI 01 - Texas Instruments SSP 10 - National Semiconductors Microwire 11 - Reserved **Value** **Description** 0x0 MOTSPI 0x1 TISSP 0x2 NATMW	RW	0x0

Bit	Name	Description	Access	Reset
3:0	dfs	Data Frame Size. Selects the data frame length. When the data frame size is programmed to be less than 16 bits, the receive data are automatically right-justified by the receive logic, with the upper bits of the receive FIFO zero-padded. You must right-justify transmit data before writing into the transmit FIFO. The transmit logic ignores the upper unused bits when transmitting the data **Value** — **Description** 0xa — WIDTH11BIT 0xb — WIDTH12BIT 0xc — WIDTH13BIT 0xd — WIDTH14BIT 0xe — WIDTH15BIT 0xf — WIDTH16BIT 0x3 — WIDTH4BIT 0x4 — WIDTH5BIT 0x5 — WIDTH6BIT 0x6 — WIDTH7BIT 0x7 — WIDTH8BIT 0x8 — WIDTH9BIT 0x9 — WIDTH10BIT	RW	0x7

ctrlr1

Control Register 1
This register exists only when the DW_apb_ssi is configured as a master device. When the DW_apb_ssi is configured as a serial slave, writing to this location has no effect; reading from this location returns 0. Control register 1 controls the end of serial transfers when in receive-only mode. It is impossible to write to this register when the DW_apb_ssi is enabled. The DW_apb_ssi is enabled and disabled by writing to the SSIENR register.

Module Instance	Base Address	Register Address
i_spim_0_spim	0xFFDA4000	0xFFDA4004
i_spim_1_spim	0xFFDA5000	0xFFDA5004

Offset: 0x4

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ndf RW 0x0															

ctrlr1 Fields

Bit	Name	Description	Access	Reset
15:0	ndf	Number of Data Frames. When TMOD = 10 or TMOD = 11, this register field sets the number of data frames to be continuously received by the DW_apb_ssi. The DW_apb_ssi continues to receive serial data until the number of data frames received is equal to this register value plus 1, which enables you to receive up to 64 KB of data in a continuous transfer. When the DW_apb_ssi is configured as a serial slave, the transfer continues for as long as the slave is selected. Therefore, this register serves no purpose and is not present when the DW_apb_ssi is configured as a serial slave.	RW	0x0

spienr

SSI Enable Register

Module Instance	Base Address	Register Address
i_spim_0_spim	0xFFDA4000	0xFFDA4008
i_spim_1_spim	0xFFDA5000	0xFFDA5008

Offset: 0x8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															spi_en RW 0x0

spienr Fields

Bit	Name	Description	Access	Reset
0	spi_en	SSI Enable. Enables and disables all DW_apb_ssi operations. When disabled, all serial transfers are halted immediately. Transmit and receive FIFO buffers are cleared when the device is disabled. It is impossible to program some of the DW_apb_ssi control registers when enabled. When disabled, the ssi_ sleep output is set (after delay) to inform the system that it is safe to remove the ssi_clk, thus saving power consumption in the system. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0

mwcr

Microwire Control Register.
This register controls the direction of the data word for the half-

duplex
Microwire serial protocol. It is impossible to write to this register when the DW_apb_ssi is enabled. The DW_apb_ssi is enabled and disabled by
writing to the SSIENR register.

Module Instance	Base Address	Register Address
i_spim_0_spim	0xFFDA4000	0xFFDA400C
i_spim_1_spim	0xFFDA5000	0xFFDA500C

Offset: 0xC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													mhs RW 0x0	mdd RW 0x0	mwmod RW 0x0

mwcr Fields

Bit	Name	Description	Access	Reset
2	mhs	Microwire Handshaking. Relevant only when the DW_apb_ssi is configured as a serial-master device. When configured as a serial slave, this bit field has no functionality. Used to enable and disable the busy/ ready handshaking interface for the Microwire protocol. When enabled, the DW_apb_ssi checks for a ready status from the target slave, after the transfer of the last data/control bit, before clearing the BUSY status in the SR register. 0: handshaking interface is disabled 1: handshaking interface is enabled **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0
1	mdd	Microwire Control. Defines the direction of the data word when the Microwire serial protocol is used. When this bit is set to 0, the data word is received by the DW_apb_ssi MacroCell from the external serial device. When this bit is set to 1, the data word is transmitted from the DW_apb_ssi MacroCell to the external serial device. **Value** — **Description** 0x0 — RXMODE 0x1 — TXMODE	RW	0x0

Bit	Name	Description	Access	Reset
0	mwmod	Microwire Transfer Mode. Defines whether the Microwire transfer is sequential or non-sequential. When sequential mode is used, only one control word is needed to transmit or receive a block of data words. When non-sequential mode is used, there must be a control word for each data word that is transmitted or received. 0: non-sequential transfer 1: sequential transfer **Value** **Description** 0x0 NONSEQ 0x1 SEQ	RW	0x0

ser

Slave Enable Register.
This register is valid only when the DW_apb_ssi is configured as a master
device. When the DW_apb_ssi is configured as a serial slave, writing to
this location has no effect; reading from this location returns 0. The register enables the individual slave select output lines from the DW_apb_ssi master. Up to 16 slave-select output pins are available on the
DW_apb_ssi master. You cannot write to this register when DW_apb_ssi is
busy and when SSI_EN = 1.

Module Instance	Base Address	Register Address
i_spim_0_spim	0xFFDA4000	0xFFDA4010
i_spim_1_spim	0xFFDA5000	0xFFDA5010

Offset: 0x10

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												ser RW 0x0			

ser Fields

Bit	Name	Description	Access	Reset
3:0	ser	Slave Select Enable Flag. Each bit in this register corresponds to a slave select line (ss_x_n]) from the DW_apb_ssi master. When a bit in this register is set (1), the corresponding slave select line from the master is activated when a serial transfer begins. It should be noted that setting or clearing bits in this register have no effect on the corresponding slave select outputs until a transfer is started. Before beginning a transfer, you should enable the bit in this register that corresponds to the slave device with which the master wants to communicate. When not operating in broadcast mode, only one bit in this field should be set. 1: Selected 0: Not Selected **Value** **Description** 0x0 NOTSELECTED 0x1 SELECTED	RW	0x0

baudr

Baud Rate Select.
This register is valid only when the DW_apb_ssi is configured as a master
device. When the DW_apb_ssi is configured as a serial slave, writing to
this location has no effect; reading from this location returns 0. The

register derives the frequency of the serial clock that regulates the data
transfer. The 16-bit field in this register defines the ssi_clk divider
value. It is impossible to write to this register when the DW_apb_ssi is
enabled. The DW_apb_ssi is enabled and disabled by writing to the SSIENR
register.

Module Instance	Base Address	Register Address
i_spim_0_spim	0xFFDA4000	0xFFDA4014
i_spim_1_spim	0xFFDA5000	0xFFDA5014

Offset: 0x14

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
sckdv RW 0x0															

baudr Fields

Bit	Name	Description	Access	Reset
15:0	sckdv	SSI Clock Divider. The LSB for this field is always set to 0 and is unaffected by a write operation, which ensures an even value is held in this register. If the value is 0, the serial output clock (sclk_out) is disabled. The frequency of the sclk_out is derived from the following equation: Fsclk_out = Fssi_clk/SCKDV where SCKDV is any even value between 2 and 65534. For example: for Fssi_clk = 3.6864MHz and SCKDV =2 Fsclk_out = 3.6864/2 = 1.8432MHz	RW	0x0

txftlr

Transmit FIFO Threshold Level.
This register controls the threshold value for the transmit FIFO memory.
The DW_apb_ssi is enabled and disabled by writing to the SSIENR register.

Module Instance	Base Address	Register Address
i_spim_0_spim	0xFFDA4000	0xFFDA4018
i_spim_1_spim	0xFFDA5000	0xFFDA5018

Offset: 0x18

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								tft RW 0x0							

txftlr Fields

Bit	Name	Description	Access	Reset
7:0	tft	Transmit FIFO Threshold. Controls the level of entries (or below) at which the transmit FIFO controller triggers an interrupt. The FIFO depth is configurable in the range 2-256; this register is sized to the number of address bits needed to access the FIFO. If you attempt to set this value greater than or equal to the depth of the FIFO, this field is not written and retains its current value. When the number of transmit FIFO entries is less than or equal to this value, the transmit FIFO empty interrupt is triggered.	RW	0x0

rxftlr

Receive FIFO Threshold level.
This register controls the threshold value for the receive FIFO memory.
The DW_apb_ssi is enabled and disabled by writing to the SSIENR register.

Module Instance	Base Address	Register Address
i_spim_0_spim	0xFFDA4000	0xFFDA401C
i_spim_1_spim	0xFFDA5000	0xFFDA501C

Offset: 0x1C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								rft RW 0x0							

rxftlr Fields

Bit	Name	Description	Access	Reset
7:0	rft	Receive FIFO Threshold. Controls the level of entries (or above) at which the receive FIFO controller triggers an interrupt. The FIFO depth is configurable in the range 2-256. This register is sized to the number of address bits needed to access the FIFO. If you attempt to set this value greater than the depth of the FIFO, this field is not written and retains its current value. When the number of receive FIFO entries is greater than or equal to this value + 1, the receive FIFO full interrupt is triggered.	RW	0x0

txflr

Transmit FIFO Level Register

Module Instance	Base Address	Register Address
i_spim_0_spim	0xFFDA4000	0xFFDA4020
i_spim_1_spim	0xFFDA5000	0xFFDA5020

Offset: 0x20

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							txtfl RO 0x0								

txflr Fields

Bit	Name	Description	Access	Reset
8:0	txtfl	Transmit FIFO Level. Contains the number of valid data entries in the transmit FIFO.	RO	0x0

rxflr

Receive FIFO Level Register

Module Instance	Base Address	Register Address
i_spim_0_spim	0xFFDA4000	0xFFDA4024
i_spim_1_spim	0xFFDA5000	0xFFDA5024

Offset: 0x24

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							rxtfl RO 0x0								

rxflr Fields

Bit	Name	Description	Access	Reset
8:0	rxtfl	Receive FIFO Level. Contains the number of valid data entries in the receive FIFO.	RO	0x0

sr

This register is used to indicate the current transfer status, FIFO status, and any transmission/reception errors that may have occurred. The status register may be read at any time. None of the bits in this register request an interrupt.

Module Instance	Base Address	Register Address
i_spim_0_spim	0xFFDA4000	0xFFDA4028
i_spim_1_spim	0xFFDA5000	0xFFDA5028

Offset: 0x28

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									dcol RO 0x0	Reserved	rff RO 0x0	rfne RO 0x0	tfe RO 0x1	tfnf RO 0x1	busy RO 0x0

sr Fields

Bit	Name	Description	Access	Reset
6	dcol	Relevant only when the DW_apb_ssi is configured as a master device. This bit is set if the DW_apb_ssi master is actively transmitting when another master selects this device as a slave. This informs the processor that the last data transfer was halted before completion. This bit is cleared when read. **Value** **Description** 0x0 NOERROR 0x1 ERROR	RO	0x0
4	rff	Reports receive FIFO condition. **Value** **Description** 0x0 NOTFULL 0x1 FULL	RO	0x0
3	rfne	Reports receive FIFO condition. **Value** **Description** 0x0 EMPTY 0x1 NOTEMPTY	RO	0x0
2	tfe	Reports transmit FIFO condition. **Value** **Description** 0x0 NOTEMPTY 0x1 EMPTY	RO	0x1
1	tfnf	Reports transmit FIFO condition. **Value** **Description** 0x0 FULL 0x1 NOTFULL	RO	0x1

Bit	Name	Description	Access	Reset
0	busy	Reports the staus of a serial transfer Value: 0x0 — INACTIVE Value: 0x1 — ACTIVE	RO	0x0

imr

Interrupt Mask Register

Module Instance	Base Address	Register Address
i_spim_0_spim	0xFFDA4000	0xFFDA402C
i_spim_1_spim	0xFFDA5000	0xFFDA502C

Offset: 0x2C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										mstim RW 0x1	rxfim RW 0x1	rxoim RW 0x1	rxuim RW 0x1	txoim RW 0x1	txeim RW 0x1

imr Fields

Bit	Name	Description	Access	Reset
5	mstim	Multi-Master Contention Interrupt Mask. This bit field is not present if the DW_apb_ssi is configured as a serial-slave device. 0 - ssi_mst_intr interrupt is masked 1 - ssi_mst_intr interrupt is not masked	RW	0x1
4	rxfim	Receive FIFO Full Interrupt Mask 0 - ssi_rxf_intr interrupt is masked 1 - ssi_rxf_intr interrupt is not masked **Value** **Description** 0x0 MASKED 0x1 ENABLED	RW	0x1
3	rxoim	Receive FIFO Overflow Interrupt Mask 0 - ssi_rxo_intr interrupt is masked 1 - ssi_rxo_intr interrupt is not masked **Value** **Description** 0x0 MASKED 0x1 ENABLED	RW	0x1
2	rxuim	Receive FIFO Underflow Interrupt Mask 0 - ssi_rxu_intr interrupt is masked 1 - ssi_rxu_intr interrupt is not masked **Value** **Description** 0x0 MASKED 0x1 ENABLED	RW	0x1

Bit	Name	Description	Access	Reset
1	txoim	Transmit FIFO Overflow Interrupt Mask 0 - ssi_txo_intr interrupt is masked 1 - ssi_txo_intr interrupt is not masked **Value** **Description** 0x0 MASKED 0x1 ENABLED	RW	0x1
0	txeim	Transmit FIFO Empty Interrupt Mask 0 - ssi_txe_intr interrupt is masked 1 - ssi_txe_intr interrupt is not masked **Value** **Description** 0x0 MASKED 0x1 ENABLED	RW	0x1

isr

Interrupt Status Register

Module Instance	Base Address	Register Address
i_spim_0_spim	0xFFDA4000	0xFFDA4030
i_spim_1_spim	0xFFDA5000	0xFFDA5030

Offset: 0x30

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										mstis RO 0x0	rxfis RO 0x0	rxois RO 0x0	rxuis RO 0x0	txois RO 0x0	txeis RO 0x0

isr Fields

Bit	Name	Description	Access	Reset
5	mstis	Multi-Master Contention Interrupt Status. This bit field is not present if the DW_apb_ssi is configured as a serial-slave device. 0 = ssi_mst_intr interrupt not active after masking 1 = ssi_mst_intr interrupt is active after masking **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0
4	rxfis	Receive FIFO Full Interrupt Status 0 = ssi_rxf_intr interrupt is not active after masking 1 = ssi_rxf_intr interrupt is full after masking **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0
3	rxois	Receive FIFO Overflow Interrupt Status 0 = ssi_rxo_intr interrupt is not active after masking 1 = ssi_rxo_intr interrupt is active after masking **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0

Bit	Name	Description	Access	Reset
2	rxuis	Receive FIFO Underflow Interrupt Status 0 = ssi_rxu_intr interrupt is not active after masking 1 = ssi_rxu_intr interrupt is active after masking **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0
1	txois	Transmit FIFO Overflow Interrupt Status 0 = ssi_txo_intr interrupt is not active after masking 1 = ssi_txo_intr interrupt is active after masking **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0
0	txeis	Transmit FIFO Empty Interrupt Status 0 = ssi_txe_intr interrupt is not active after masking 1 = ssi_txe_intr interrupt is active after masking **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0

risr

Raw Interrupt StatusRegister

Module Instance	Base Address	Register Address
i_spim_0_spim	0xFFDA4000	0xFFDA4034
i_spim_1_spim	0xFFDA5000	0xFFDA5034

Offset: 0x34

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										mstir RO 0x0	rxfir RO 0x0	rxoir RO 0x0	rxuir RO 0x0	txoir RO 0x0	txeir RO 0x0

risr Fields

Bit	Name	Description	Access	Reset
5	mstir	Multi-Master Contention Raw Interrupt Status. This bit field is not present if the DW_apb_ssi is configured as a serial-slave device. 0 = ssi_mst_intr interrupt is not active prior to masking 1 = ssi_mst_intr interrupt is active prior masking	RO	0x0
4	rxfir	Receive FIFO Full Raw Interrupt Status 0 = ssi_rxf_intr interrupt is not active prior to masking 1 = ssi_rxf_intr interrupt is active prior to masking **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0

Bit	Name	Description	Access	Reset
3	rxoir	Receive FIFO Overflow Raw Interrupt Status 0 = ssi_rxo_intr interrupt is not active prior to masking 1 = ssi_rxo_intr interrupt is active prior masking **Value** **Description** 0x0 INACTOVE 0x1 ACTIVE	RO	0x0
2	rxuir	Receive FIFO Underflow Raw Interrupt Status 0 = ssi_rxu_intr interrupt is not active prior to masking 1 = ssi_rxu_intr interrupt is active prior to masking **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0
1	txoir	Transmit FIFO Overflow Raw Interrupt Status 0 = ssi_txo_intr interrupt is not active prior to masking 1 = ssi_txo_intr interrupt is active prior masking **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0
0	txeir	Transmit FIFO Empty Raw Interrupt Status 0 = ssi_txe_intr interrupt is not active prior to masking 1 = ssi_txe_intr interrupt is active prior masking **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0

txoicr

Transmit FIFO Overflow Interrupt Clear Register

Module Instance	Base Address	Register Address
i_spim_0_spim	0xFFDA4000	0xFFDA4038
i_spim_1_spim	0xFFDA5000	0xFFDA5038

Offset: 0x38

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															txoicr RO 0x0

txoicr Fields

Bit	Name	Description	Access	Reset
0	txoicr	Clear Transmit FIFO Overflow Interrupt. This register reflects the status of the interrupt. A read from this register clears the ssi_txo_intr interrupt; writing has no effect.	RO	0x0

rxoicr

Receive FIFO Overflow Interrupt Clear Register

Module Instance	Base Address	Register Address
i_spim_0_spim	0xFFDA4000	0xFFDA403C
i_spim_1_spim	0xFFDA5000	0xFFDA503C

Offset: 0x3C

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															rxoicr RO 0x0

rxoicr Fields

Bit	Name	Description	Access	Reset
0	rxoicr	Clear Receive FIFO Overflow Interrupt. This register reflects the status of the interrupt. A read from this register clears the ssi_rxo_intr interrupt; writing has no effect.	RO	0x0

rxuicr

Receive FIFO Underflow Interrupt Clear Register

Module Instance	Base Address	Register Address
i_spim_0_spim	0xFFDA4000	0xFFDA4040
i_spim_1_spim	0xFFDA5000	0xFFDA5040

Offset: 0x40

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															rxuicr RO 0x0

rxuicr Fields

Bit	Name	Description	Access	Reset
0	rxuicr	Clear Receive FIFO Underflow Interrupt. This register reflects the status of the interrupt. A read from this register clears the ssi_rxu_intr interrupt; writing has no effect.	RO	0x0

icr

Interrupt Clear Register

Module Instance	Base Address	Register Address
i_spim_0_spim	0xFFDA4000	0xFFDA4048
i_spim_1_spim	0xFFDA5000	0xFFDA5048

Offset: 0x48

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															icr RO 0x0

icr Fields

Bit	Name	Description	Access	Reset
0	icr	Clear Interrupts. This register is set if any of the interrupts below are active. A read clears the ssi_txo_intr, ssi_rxu_intr, ssi_rxo_intr, and the ssi_mst_intr interrupts. Writing to this register has no effect.	RO	0x0

dmacr

DMA Control Register.
This register is only valid when DW_apb_ssi is configured with a set of
DMA Controller interface signals (SSI_HAS_DMA = 1). When DW_apb_ssi is not configured for DMA operation, this register will not exist and writing
to the register's address will have no effect; reading from this register
address will return zero. The register is used to enable the DMA Controller interface operation.

Module Instance	Base Address	Register Address
i_spim_0_spim	0xFFDA4000	0xFFDA404C
i_spim_1_spim	0xFFDA5000	0xFFDA504C

Offset: 0x4C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														tdmae RW 0x0	rdmae RW 0x0

dmacr Fields

Bit	Name	Description	Access	Reset
1	tdmae	Transmit DMA Enable. This bit enables/disables the transmit FIFO DMA channel. 0 = Transmit DMA disabled 1 = Transmit DMA enabled **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
0	rdmae	Receive DMA Enable. This bit enables/disables the receive FIFO DMA channel 0 = Receive DMA disabled 1 = Receive DMA enabled **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0

dmatdlr

```
        DMA Transmit Data Level.
This register is only valid when the DW_apb_ssi is configured with a
set
of DMA interface signals (SSI_HAS_DMA = 1). When DW_apb_ssi is not
configured for DMA operation, this register will not exist and writing
to its address will have no effect; reading from its address will
return zero.
```

Module Instance	Base Address	Register Address
i_spim_0_spim	0xFFDA4000	0xFFDA4050
i_spim_1_spim	0xFFDA5000	0xFFDA5050

Offset: 0x50

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								dmatdl RW 0x0							

dmatdlr Fields

Bit	Name	Description	Access	Reset
7:0	dmatdl	Transmit Data Level. This bit field controls the level at which a DMA request is made by the transmit logic. It is equal to the watermark level; that is, the dma_tx_req signal is generated when the number of valid data entries in the transmit FIFO is equal to or below this field value, and TDMAE = 1.	RW	0x0

dmardlr

DMA Receive Data Level.
This register is only valid when DW_apb_ssi is configured with a set of
DMA interface signals (SSI_HAS_DMA = 1). When DW_apb_ssi is not configured
for DMA operation, this register will not exist and writing to its address
will have no effect; reading from its address will return zero.

Module Instance	Base Address	Register Address
i_spim_0_spim	0xFFDA4000	0xFFDA4054
i_spim_1_spim	0xFFDA5000	0xFFDA5054

Offset: 0x54

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								dmardl RW 0x0							

dmardlr Fields

Bit	Name	Description	Access	Reset
7:0	dmardl	Receive Data Level. This bit field controls the level at which a DMA request is made by the receive logic. The watermark level = DMARDL+1; that is, dma_rx_req is generated when the number of valid data entries in the receive FIFO is equal to or above this field value + 1, and RDMAE=1.	RW	0x0

idr

Identification Register.
This register contains the peripherals identification code, which is written into the register at configuration time using coreConsultant.

Module Instance	Base Address	Register Address
i_spim_0_spim	0xFFDA4000	0xFFDA4058
i_spim_1_spim	0xFFDA5000	0xFFDA5058

Offset: 0x58

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
idr RO 0x5510000															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
idr RO 0x5510000															

idr Fields

Bit	Name	Description	Access	Reset
31:0	idr	This register contains the peripherals identification code	RO	0x5510000

spi_version_id

coreKit Version ID Register

Module Instance	Base Address	Register Address
i_spim_0_spim	0xFFDA4000	0xFFDA405C
i_spim_1_spim	0xFFDA5000	0xFFDA505C

Offset: 0x5C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
spi_version_id RW 0x3332322A															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
spi_version_id RW 0x3332322A															

spi_version_id Fields

Bit	Name	Description	Access	Reset
31:0	spi_version_id	Contains the hex representation of the Synopsys component version. Consists of ASCII value for each number in the version.	RW	0x3332322A

dr

The data register is a 16-bit read/write buffer for the transmit/receive FIFOs. When the register is read, data in the receive FIFO buffer is accessed. When it is written to, data are moved into the transmit FIFO buffer; a write can occur only when SSI_EN = 1. FIFOs are reset when SSI_EN = 0.Note that the data register occupies thirty-six 32-bit address locations of the memory map to facilitate burst transfers. Writing to any of these address locations has the same effect as pushing the data from the bus into the TxFIFO and reading from any of these location has the same effect as popping data from the RxFIFO onto the bus.

Module Instance	Base Address	Register Address
i_spim_0_spim	0xFFDA4000	0xFFDA4060 to 0xFFDA40EC
i_spim_1_spim	0xFFDA5000	0xFFDA5060 to 0xFFDA50EC

Offset: 0x60 to 0xEC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
dr RW 0x0															

dr Fields

Bit	Name	Description	Access	Reset
15:0	dr	Data Register. When writing to this register, you must right-justify the data. Read data are automatically right-justified. Read = Receive FIFO buffer Write = Transmit FIFO buffer	RW	0x0

rx_sample_dly

RX Sample Delay.
This register controls the number of ssi_clk cycles that are delayed (from the default sample time) before the actual sample of the rxd input occurs. It is impossible to write to this register when the SPI master is enabled. The SPI master is enabled and disabled by writing to the SSIENR register.

Module Instance	Base Address	Register Address
i_spim_0_spim	0xFFDA4000	0xFFDA40F0
i_spim_1_spim	0xFFDA5000	0xFFDA50F0

Offset: 0xF0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								rsd RW 0x0							

rx_sample_dly Fields

Bit	Name	Description	Access	Reset
7:0	rsd	Rxd Sample Delay. This register is used to delay the sample of the rxd input port. Each value represents a single ssi_clk delay on the sample of rxd. NOTE: You must program the rsd field to 0x1 or greater for correct operation. In addition, if you program this register to a value that exceeds the depth of the internal shift registers (SSI_RX_DLY_SR_DEPTH), zero delay is applied to the rxd sample and the interface does not function correctly.	RW	0x0

msticr

Multi-Master Interrupt Clear Register

Module Instance	Base Address	Register Address
i_spim_0_spim	0xFFDA4000	0xFFDA4044
i_spim_1_spim	0xFFDA5000	0xFFDA5044

Offset: 0x44

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															msticr RO 0x0

msticr Fields

Bit	Name	Description	Access	Reset
0	msticr	Clear Multi-Master Contention Interrupt. This register reflects the status of the interrupt. A read from this register clears the ssi_mst_intr interrupt; writing has no effect.	RO	0x0

rsvd_0

RSVD_0 - Reserved address location

Module Instance	Base Address	Register Address
i_spim_0_spim	0xFFDA4000	0xFFDA40F4
i_spim_1_spim	0xFFDA5000	0xFFDA50F4

Offset: 0xF4

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															

rsvd_1

RSVD_1 - Reserved address location

Module Instance	Base Address	Register Address
i_spim_0_spim	0xFFDA4000	0xFFDA40F8
i_spim_1_spim	0xFFDA5000	0xFFDA50F8

Offset: 0xF8

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															

rsvd_2

RSVD_2 - Reserved address location

Module Instance	Base Address	Register Address
i_spim_0_spim	0xFFDA4000	0xFFDA40FC
i_spim_1_spim	0xFFDA5000	0xFFDA50FC

Offset: 0xFC

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															

spis Address Map

Module Instance	Base Address	End Address
i_spis_0_spis	0xFFDA2000	0xFFDA2FFF

Module Instance	Base Address	End Address
i_spis_1_spis	0xFFDA3000	0xFFDA3FFF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
ctrlr0 on page 19-100	0x0	32	RW	0x7	Control Register 0: This register controls the serial data transfer. It is impossible to write to this register when the DW_apb_ssi is enabled. The DW_apb_ssi is enabled and disabled by writing to the SSIENR register.
spienr on page 19-106	0x8	32	RW	0x0	SSI Enable Register
mwcr on page 19-107	0xC	32	RW	0x0	Microwire Control Register. This register controls the direction of the data word for the half-duplex Microwire serial protocol. It is impossible to write to this register when the DW_apb_ssi is enabled. The DW_apb_ssi is enabled and disabled by writing to the SSIENR register.
txftlr on page 19-109	0x18	32	RW	0x0	Transmit FIFO Threshold Level. This register controls the threshold value for the transmit FIFO memory. The DW_apb_ssi is enabled and disabled by writing to the SSIENR register.

Register	Offset	Width	Access	Reset Value	Description
rxftlr on page 19-110	0x1C	32	RW	0x0	Receive FIFO Threshold level. This register controls the threshold value for the receive FIFO memory. The DW_apb_ssi is enabled and disabled by writing to the SSIENR register.
txflr on page 19-111	0x20	32	RO	0x0	Transmit FIFO Level Register
rxflr on page 19-112	0x24	32	RO	0x0	Receive FIFO Level Register
sr on page 19-113	0x28	32	RO	0x6	Status Register. This is a read-only register used to indicate the current transfer status, FIFO status, and any transmission/reception errors that may have occurred. The status register may be read at any time. None of the bits in this register request an interrupt.
imr on page 19-115	0x2C	32	RW	0x1F	Interrupt Mask Register
isr on page 19-117	0x30	32	RO	0x0	Interrupt Status Register
risr on page 19-119	0x34	32	RO	0x0	Raw Interrupt StatusRegister
txoicr on page 19-121	0x38	32	RO	0x0	Transmit FIFO Overflow Interrupt Clear Register
rxoicr on page 19-122	0x3C	32	RO	0x0	Receive FIFO Overflow Interrupt Clear Register

Register	Offset	Width	Access	Reset Value	Description
rxuicr on page 19-123	0x40	32	RO	0x0	Receive FIFO Underflow Interrupt Clear Register
icr on page 19-124	0x48	32	RO	0x0	Interrupt Clear Register
dmacr on page 19-125	0x4C	32	RW	0x0	DMA Control Register. This register is only valid when DW_apb_ssi is configured with a set of DMA Controller interface signals (SSI_HAS_DMA = 1). When DW_apb_ssi is not configured for DMA operation, this register will not exist and writing to the register's address will have no effect; reading from this register address will return zero. The register is used to enable the DMA Controller interface operation.
dmatdlr on page 19-126	0x50	32	RW	0x0	DMA Transmit Data Level. This register is only valid when the DW_apb_ssi is configured with a set of DMA interface signals (SSI_HAS_DMA = 1). When DW_apb_ssi is not configured for DMA operation, this register will not exist and writing to its address will have no effect; reading from its address will return zero.

Register	Offset	Width	Access	Reset Value	Description
dmardlr on page 19-127	0x54	32	RW	0x0	DMA Receive Data Level. This register is only valid when DW_apb_ssi is configured with a set of DMA interface signals (SSI_HAS_DMA = 1). When DW_apb_ssi is not configured for DMA operation, this register will not exist and writing to its address will have no effect; reading from its address will return zero.
idr on page 19-128	0x58	32	RO	0x5510005	Identification Register. This register contains the peripherals identification code, which is written into the register at configura-tion time using coreConsultant.
spi_version_id on page 19-129	0x5C	32	RW	0x3332322A	coreKit Version ID Register
dr on page 19-130	0x60	32	RW	0x0	The DW_apb_ssi data register is a 16-bit read/write buffer for the transmit/receive FIFOs. When the register is read, data in the receive FIFO buffer is accessed. When it is written to, data are moved into the transmit FIFO buffer; a write can occur only when SSI_EN = 1. FIFOs are reset when SSI_EN = 0.

Register	Offset	Width	Access	Reset Value	Description
msticr on page 19-131	0x44	32	RO	0x0	Multi-Master Interrupt Clear Register

spis Summary

Module Instance	Base Address
i_spis_0_spis	0xFFDA2000
i_spis_1_spis	0xFFDA3000

Register Address Offset	Bit Fields															
i_spis_0_spis																
ctrlr0 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cfs RW 0x0				srl RW 0x0	slv_oe RW 0x0	tmod RW 0x0		scpol RW 0x0	scph RW 0x0	frf RW 0x0		dfs RW 0x7			
spienr 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															spi_en RW 0x0
mwcr 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														mdd RW 0x0	mwmod RW 0x0
txftlr 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								tft RW 0x0							

Register Address Offset	Bit Fields															
rxftlr 0x1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								rft RW 0x0							
txflr 0x20	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							txtfl RO 0x0								
rxflr 0x24	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							rxtfl RO 0x0								
sr 0x28	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved										txe RO 0x0	rff RO 0x0	rfne RO 0x0	tfe RO 0x1	tfnf RO 0x1	busy RO 0x0
imr 0x2C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved											rxfim RW 0x1	rxoim RW 0x1	rxuim RW 0x1	txoim RW 0x1	txeim RW 0x1
isr 0x30	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved											rxfis RO 0x0	rxois RO 0x0	rxuis RO 0x0	txois RO 0x0	txeis RO 0x0

Register Address Offset	Bit Fields															
risr 0x34	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved											rxfir RO 0x0	rxoir RO 0x0	rxuir RO 0x0	txoir RO 0x0	txeir RO 0x0
txoicr 0x38	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															txoicr RO 0x0
rxoicr 0x3C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															rxoicr RO 0x0
rxuicr 0x40	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															rxuicr RO 0x0
icr 0x48	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															icr RO 0x0
dmacr 0x4C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														tdmae RW 0x0	rdmae RW 0x0

Register Address Offset	Bit Fields															
dmatdlr 0x50	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								dmatdl RW 0x0							
dmardlr 0x54	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								dmardl RW 0x0							
idr 0x58	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	idr RO 0x5510005															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	idr RO 0x5510005															
spi_version_id 0x5C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	spi_version_id RW 0x3332322A															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	spi_version_id RW 0x3332322A															
dr 0x60	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	dr RW 0x0															
msticr 0x44	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															msticr RO 0x0
i_spis_1_spis																

Register Address Offset	Bit Fields															
ctrlr0 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cfs RW 0x0				srl RW 0x0	slv_oe RW 0x0	tmod RW 0x0		scpol RW 0x0	scph RW 0x0	frf RW 0x0		dfs RW 0x7			
spienr 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															spi_en RW 0x0
mwcr 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														mdd RW 0x0	mwmod RW 0x0
txftlr 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								tft RW 0x0							
rxftlr 0x1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								rft RW 0x0							
txflr 0x20	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							txtfl RO 0x0								

Register Address Offset	Bit Fields															
rxflr 0x24	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							rxtfl RO 0x0								
sr 0x28	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved										txe RO 0x0	rff RO 0x0	rfne RO 0x0	tfe RO 0x1	tfnf RO 0x1	busy RO 0x0
imr 0x2C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved											rxfim RW 0x1	rxoim RW 0x1	rxuim RW 0x1	txoim RW 0x1	txeim RW 0x1
isr 0x30	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved											rxfis RO 0x0	rxois RO 0x0	rxuis RO 0x0	txois RO 0x0	txeis RO 0x0
risr 0x34	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved											rxfir RO 0x0	rxoir RO 0x0	rxuir RO 0x0	txoir RO 0x0	txeir RO 0x0
txoicr 0x38	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															txoicr RO 0x0

Register Address Offset	Bit Fields															
rxoicr 0x3C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															rxoicr RO 0x0
rxuicr 0x40	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															rxuicr RO 0x0
icr 0x48	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															icr RO 0x0
dmacr 0x4C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														tdmae RW 0x0	rdmae RW 0x0
dmatdlr 0x50	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								dmatdl RW 0x0							
dmardlr 0x54	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								dmardl RW 0x0							

Register Address Offset	Bit Fields															
idr 0x58	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	idr RO 0x5510005															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	idr RO 0x5510005															
spi_version_id 0x5C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	spi_version_id RW 0x3332322A															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	spi_version_id RW 0x3332322A															
dr 0x60	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	dr RW 0x0															
msticr 0x44	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															msticr RO 0x0

ctrlr0

Control Register 0:
This register controls the serial data transfer. It is impossible to write to this register when the DW_apb_ssi is enabled. The DW_apb_ssi is enabled and disabled by writing to the SSIENR register.

Module Instance	Base Address	Register Address
i_spis_0_spis	0xFFDA2000	0xFFDA2000
i_spis_1_spis	0xFFDA3000	0xFFDA3000

Offset: 0x0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cfs RW 0x0				srl RW 0x0	slv_oe RW 0x0	tmod RW 0x0		scpol RW 0x0	scph RW 0x0	frf RW 0x0		dfs RW 0x7			

ctrlr0 Fields

Bit	Name	Description	Access	Reset
15:12	cfs	Control Frame Size. Selects the length of the control word for the Microwire frame format Value — Description 0xa — SIZE11BIT 0xb — SIZE12BIT 0xc — SIZE13BIT 0xd — SIZE14BIT 0xe — SIZE15BIT 0xf — SIZE16BIT 0x0 — SIZE1BIT 0x1 — SIZE2BIT 0x2 — SIZE3BIT 0x3 — SIZE4BIT 0x4 — SIZE5BIT 0x5 — SIZE6BIT 0x6 — SIZE7BIT 0x7 — SIZE8BIT 0x8 — SIZE9BIT 0x9 — SIZE10BIT	RW	0x0

Bit	Name	Description	Access	Reset
11	srl	Shift Register Loop. Used for testing purposes only. When internally active, connects the transmit shift register output to the receive shift register input. 0 - Normal Mode Operation 1 - Test Mode Operation **Value** / **Description** 0x0 NORMMODE 0x1 TESTMODE	RW	0x0
10	slv_oe	Slave Output Enable. Relevant only when the DW_apb_ssi is configured as a serial-slave device. When configured as a serial master, this bit field has no functionality. This bit enables or disables the setting of the ssi_oe_n output from the DW_apb_ssi serial slave. When SLV_OE = 1, the ssi_oe_n output can never be active. When the ssi_oe_n output controls the tri-state buffer on the txd output from the slave, a high impedance state is always present on the slave txd output when SLV_OE = 1. This is useful when the master transmits in broadcast mode (master transmits data to all slave devices). Only one slave may respond with data on the master rxd line. This bit is enabled after reset and must be disabled by software (when broadcast mode is used), if you do not want this device to respond with data. 0 - Slave txd is enabled 1 - Slave txd is disabled **Value** / **Description** 0x0 ENABLED 0x1 DISABLED	RW	0x0

Bit	Name	Description	Access	Reset
9:8	tmod	Transfer Mode. Selects the mode of transfer for serial communication. This field does not affect the transfer duplicity. Only indicates whether the receive or transmit data are valid. In transmit-only mode, data received from the external device is not valid and is not stored in the receive FIFO memory; it is overwritten on the next transfer. In receive-only mode, transmitted data are not valid. After the first write to the transmit FIFO, the same word is retransmitted for the duration of the transfer. In transmit-and-receive mode, both transmit and receive data are valid. The transfer continues until the transmit FIFO is empty. Data received from the external device are stored into the receive FIFO memory, where it can be accessed by the host processor. 00 - Transmit & Receive 01 - Transmit Only 10 - Receive Only 11 - Reserved **Value** **Description** 0x0 TXRX 0x1 TXONLY 0x2 RXONLY	RW	0x0

Bit	Name	Description	Access	Reset
7	scpol	Serial Clock Polarity. Valid when the frame format (FRF) is set to Motorola SPI. Used to select the polarity of the inactive serial clock, which is held inactive when the DW_apb_ssi master is not actively transferring data on the serial bus. 0 - Inactive state of serial clock is low 1 - Inactive state of serial clock is high **Value** **Description** 0x0 INACTIVELOW 0x1 INACTIVEHIGH	RW	0x0
6	scph	Serial Clock Phase. Valid when the frame format (FRF) is set to Motorola SPI. The serial clock phase selects the relation-ship of the serial clock with the slave select signal. When SCPH = 0, data are captured on the first edge of the serial clock. When SCPH = 1, the serial clock starts toggling one cycle after the slave select line is activated, and data are captured on the second edge of the serial clock. 0: Serial clock toggles in middle of first data bit 1: Serial clock toggles at start of first data bit **Value** **Description** 0x0 MIDBIT 0x1 STARTBIT	RW	0x0

Bit	Name	Description	Access	Reset
5:4	frf	Frame Format. Selects which serial protocol transfers the data. 00 - Motorola SPI 01 - Texas Instruments SSP 10 - National Semiconductors Microwire 11 - Reserved **Value** **Description** 0x0 MOTSPI 0x1 TISSP 0x2 NATMW	RW	0x0

Bit	Name	Description	Access	Reset
3:0	dfs	Data Frame Size. Selects the data frame length. When the data frame size is programmed to be less than 16 bits, the receive data are automatically right-justified by the receive logic, with the upper bits of the receive FIFO zero-padded. You must right-justify transmit data before writing into the transmit FIFO. The transmit logic ignores the upper unused bits when transmitting the data **Value** — **Description** 0xa — WIDTH11BIT 0xb — WIDTH12BIT 0xc — WIDTH13BIT 0xd — WIDTH14BIT 0xe — WIDTH15BIT 0xf — WIDTH16BIT 0x3 — WIDTH4BIT 0x4 — WIDTH5BIT 0x5 — WIDTH6BIT 0x6 — WIDTH7BIT 0x7 — WIDTH8BIT 0x8 — WIDTH9BIT 0x9 — WIDTH10BIT	RW	0x7

spienr

SSI Enable Register

Module Instance	Base Address	Register Address
i_spis_0_spis	0xFFDA2000	0xFFDA2008
i_spis_1_spis	0xFFDA3000	0xFFDA3008

Offset: 0x8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															spi_en RW 0x0

spienr Fields

Bit	Name	Description	Access	Reset
0	spi_en	SSI Enable. Enables and disables all DW_apb_ssi operations. When disabled, all serial transfers are halted immediately. Transmit and receive FIFO buffers are cleared when the device is disabled. It is impossible to program some of the DW_apb_ssi control registers when enabled. When disabled, the ssi_ sleep output is set (after delay) to inform the system that it is safe to remove the ssi_clk, thus saving power consumption in the system. **Value** — **Description** 0x0 — DISABLED 0x1 — ENABLED	RW	0x0

mwcr

```
        Microwire Control Register.
This register controls the direction of the data word for the half-
duplex
Microwire serial protocol. It is impossible to write to this register
when the DW_apb_ssi is enabled. The DW_apb_ssi is enabled and
disabled by
writing to the SSIENR register.
```

Module Instance	Base Address	Register Address
i_spis_0_spis	0xFFDA2000	0xFFDA200C
i_spis_1_spis	0xFFDA3000	0xFFDA300C

Offset: 0xC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														mdd RW 0x0	mwmod RW 0x0

mwcr Fields

Bit	Name	Description	Access	Reset
1	mdd	Microwire Control. Defines the direction of the data word when the Microwire serial protocol is used. When this bit is set to 0, the data word is received by the DW_apb_ssi MacroCell from the external serial device. When this bit is set to 1, the data word is transmitted from the DW_apb_ssi MacroCell to the external serial device. **Value** **Description** 0x0 RXMODE 0x1 TXMODE	RW	0x0

Bit	Name	Description	Access	Reset
0	mwmod	Microwire Transfer Mode. Defines whether the Microwire transfer is sequential or non-sequential. When sequential mode is used, only one control word is needed to transmit or receive a block of data words. When non-sequential mode is used, there must be a control word for each data word that is transmitted or received. 0: non-sequential transfer 1: sequential transfer **Value** / **Description** 0x0 NONSEQ 0x1 SEQ	RW	0x0

txftlr

Transmit FIFO Threshold Level.
This register controls the threshold value for the transmit FIFO memory.
The DW_apb_ssi is enabled and disabled by writing to the SSIENR register.

Module Instance	Base Address	Register Address
i_spis_0_spis	0xFFDA2000	0xFFDA2018
i_spis_1_spis	0xFFDA3000	0xFFDA3018

Offset: 0x18

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								tft RW 0x0							

txftlr Fields

Bit	Name	Description	Access	Reset
7:0	tft	Transmit FIFO Threshold. Controls the level of entries (or below) at which the transmit FIFO controller triggers an interrupt. The FIFO depth is configurable in the range 2-256; this register is sized to the number of address bits needed to access the FIFO. If you attempt to set this value greater than or equal to the depth of the FIFO, this field is not written and retains its current value. When the number of transmit FIFO entries is less than or equal to this value, the transmit FIFO empty interrupt is triggered.	RW	0x0

rxftlr

Receive FIFO Threshold level.
This register controls the threshold value for the receive FIFO memory.
The DW_apb_ssi is enabled and disabled by writing to the SSIENR register.

Module Instance	Base Address	Register Address
i_spis_0_spis	0xFFDA2000	0xFFDA201C
i_spis_1_spis	0xFFDA3000	0xFFDA301C

Offset: 0x1C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								rft RW 0x0							

rxftlr Fields

Bit	Name	Description	Access	Reset
7:0	rft	Receive FIFO Threshold. Controls the level of entries (or above) at which the receive FIFO controller triggers an interrupt. The FIFO depth is configurable in the range 2-256. This register is sized to the number of address bits needed to access the FIFO. If you attempt to set this value greater than the depth of the FIFO, this field is not written and retains its current value. When the number of receive FIFO entries is greater than or equal to this value + 1, the receive FIFO full interrupt is triggered.	RW	0x0

txflr

Transmit FIFO Level Register

Module Instance	Base Address	Register Address
i_spis_0_spis	0xFFDA2000	0xFFDA2020
i_spis_1_spis	0xFFDA3000	0xFFDA3020

Offset: 0x20

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							txtfl RO 0x0								

txflr Fields

Bit	Name	Description	Access	Reset
8:0	txtfl	Transmit FIFO Level. Contains the number of valid data entries in the transmit FIFO.	RO	0x0

rxflr

Receive FIFO Level Register

Module Instance	Base Address	Register Address
i_spis_0_spis	0xFFDA2000	0xFFDA2024
i_spis_1_spis	0xFFDA3000	0xFFDA3024

Offset: 0x24

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							rxtfl RO 0x0								

rxflr Fields

Bit	Name	Description	Access	Reset
8:0	rxtfl	Receive FIFO Level. Contains the number of valid data entries in the receive FIFO.	RO	0x0

sr

Status Register.
This is a read-only register used to indicate the current transfer status,
FIFO status, and any transmission/reception errors that may have occurred.
The status register may be read at any time. None of the bits in this register request an interrupt.

Module Instance	Base Address	Register Address
i_spis_0_spis	0xFFDA2000	0xFFDA2028
i_spis_1_spis	0xFFDA3000	0xFFDA3028

Offset: 0x28

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										txe RO 0x0	rff RO 0x0	rfne RO 0x0	tfe RO 0x1	tfnf RO 0x1	busy RO 0x0

sr Fields

Bit	Name	Description	Access	Reset
5	txe	Transmission Error. Set if the transmit FIFO is empty when a transfer is started. This bit can be set only when the DW_apb_ssi is configured as a slave device. Datafrom the previous transmission is resent on the txd line. This bit is cleared when read. 0 - No error 1 - Transmission error **Value: Description** 0x0: NOERROR 0x1: ERROR	RO	0x0
4	rff	Receive FIFO Full. When the receive FIFO is completely full, this bit is set. When the receive FIFO contains one or more empty location, this bit is cleared. 0 - Receive FIFO is not full 1 - Receive FIFO is full **Value: Description** 0x0: NOTFULL 0x1: FULL	RO	0x0
3	rfne	Receive FIFO Not Empty. Set when the receive FIFO contains one or more entries and is cleared when the receive FIFO is empty. This bit can be polled by software to completely empty the receive FIFO. 0 - Receive FIFO is empty 1 - Receive FIFO is not empty **Value: Description** 0x0: EMPTY 0x1: NOTEMPTY	RO	0x0

Bit	Name	Description	Access	Reset
2	tfe	Transmit FIFO Empty. When the transmit FIFO is completely empty, this bit is set. When the transmit FIFO contains one or more valid entries, this bit is cleared. This bit field does not request an interrupt. 0 - Transmit FIFO is not empty 1 - Transmit FIFO is empty **Value** / **Description** 0x0 NOTEMPTY 0x1 EMPTY	RO	0x1
1	tfnf	Transmit FIFO Not Full. Set when the transmit FIFO contains one or more empty locations, and is cleared when the FIFO is full. 0 - Transmit FIFO is full 1 - Transmit FIFO is not full **Value** / **Description** 0x0 FULL 0x1 NOTFULL	RO	0x1
0	busy	SSI Busy Flag. When set, indicates that a serial transfer is in progress; when cleared indicates that the DW_apb_ssi is idle or disabled. 0 - DW_apb_ssi is idle or disabled 1 - DW_apb_ssi is actively transfer-ring data **Value** / **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0

imr

Interrupt Mask Register

Module Instance	Base Address	Register Address
i_spis_0_spis	0xFFDA2000	0xFFDA202C
i_spis_1_spis	0xFFDA3000	0xFFDA302C

Offset: 0x2C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved											rxfim RW 0x1	rxoim RW 0x1	rxuim RW 0x1	txoim RW 0x1	txeim RW 0x1

imr Fields

Bit	Name	Description	Access	Reset
4	rxfim	Receive FIFO Full Interrupt Mask 0 - ssi_rxf_intr interrupt is masked 1 - ssi_rxf_intr interrupt is not masked **Value** **Description** 0x0 MASKED 0x1 ENABLED	RW	0x1
3	rxoim	Receive FIFO Overflow Interrupt Mask 0 - ssi_rxo_intr interrupt is masked 1 - ssi_rxo_intr interrupt is not masked **Value** **Description** 0x0 MASKED 0x1 ENABLED	RW	0x1

Bit	Name	Description	Access	Reset
2	rxuim	Receive FIFO Underflow Interrupt Mask 0 - ssi_rxu_intr interrupt is masked 1 - ssi_rxu_intr interrupt is not masked **Value** **Description** 0x0 MASKED 0x1 ENABLED	RW	0x1
1	txoim	Transmit FIFO Overflow Interrupt Mask 0 - ssi_txo_intr interrupt is masked 1 - ssi_txo_intr interrupt is not masked **Value** **Description** 0x0 MASKED 0x1 ENABLED	RW	0x1
0	txeim	Transmit FIFO Empty Interrupt Mask 0 - ssi_txe_intr interrupt is masked 1 - ssi_txe_intr interrupt is not masked **Value** **Description** 0x0 MASKED 0x1 ENABLED	RW	0x1

isr

Interrupt Status Register

Module Instance	Base Address	Register Address
i_spis_0_spis	0xFFDA2000	0xFFDA2030
i_spis_1_spis	0xFFDA3000	0xFFDA3030

Offset: 0x30

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved											rxfis RO 0x0	rxois RO 0x0	rxuis RO 0x0	txois RO 0x0	txeis RO 0x0

isr Fields

Bit	Name	Description	Access	Reset
4	rxfis	Receive FIFO Full Interrupt Status 0 = ssi_rxf_intr interrupt is not active after masking 1 = ssi_rxf_intr interrupt is full after masking **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0
3	rxois	Receive FIFO Overflow Interrupt Status 0 = ssi_rxo_intr interrupt is not active after masking 1 = ssi_rxo_intr interrupt is active after masking **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0

Bit	Name	Description	Access	Reset
2	rxuis	Receive FIFO Underflow Interrupt Status 0 = ssi_rxu_intr interrupt is not active after masking 1 = ssi_rxu_intr interrupt is active after masking **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0
1	txois	Transmit FIFO Overflow Interrupt Status 0 = ssi_txo_intr interrupt is not active after masking 1 = ssi_txo_intr interrupt is active after masking **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0
0	txeis	Transmit FIFO Empty Interrupt Status 0 = ssi_txe_intr interrupt is not active after masking 1 = ssi_txe_intr interrupt is active after masking **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0

risr

Raw Interrupt StatusRegister

Module Instance	Base Address	Register Address
i_spis_0_spis	0xFFDA2000	0xFFDA2034
i_spis_1_spis	0xFFDA3000	0xFFDA3034

Offset: 0x34

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved											rxfir RO 0x0	rxoir RO 0x0	rxuir RO 0x0	txoir RO 0x0	txeir RO 0x0

risr Fields

Bit	Name	Description	Access	Reset
4	rxfir	Receive FIFO Full Raw Interrupt Status 0 = ssi_rxf_intr interrupt is not active prior to masking 1 = ssi_rxf_intr interrupt is active prior to masking **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0
3	rxoir	Receive FIFO Overflow Raw Interrupt Status 0 = ssi_rxo_intr interrupt is not active prior to masking 1 = ssi_rxo_intr interrupt is active prior masking **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0

Bit	Name	Description	Access	Reset
2	rxuir	Receive FIFO Underflow Raw Interrupt Status 0 = ssi_rxu_intr interrupt is not active prior to masking 1 = ssi_rxu_intr interrupt is active prior to masking **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0
1	txoir	Transmit FIFO Overflow Raw Interrupt Status 0 = ssi_txo_intr interrupt is not active prior to masking 1 = ssi_txo_intr interrupt is active prior masking **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0
0	txeir	Transmit FIFO Empty Raw Interrupt Status 0 = ssi_txe_intr interrupt is not active prior to masking 1 = ssi_txe_intr interrupt is active prior masking **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0

txoicr

Transmit FIFO Overflow Interrupt Clear Register

Module Instance	Base Address	Register Address
i_spis_0_spis	0xFFDA2000	0xFFDA2038
i_spis_1_spis	0xFFDA3000	0xFFDA3038

Offset: 0x38

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															txoicr RO 0x0

txoicr Fields

Bit	Name	Description	Access	Reset
0	txoicr	Clear Transmit FIFO Overflow Interrupt. This register reflects the status of the interrupt. A read from this register clears the ssi_txo_intr interrupt; writing has no effect.	RO	0x0

rxoicr

Receive FIFO Overflow Interrupt Clear Register

Module Instance	Base Address	Register Address
i_spis_0_spis	0xFFDA2000	0xFFDA203C
i_spis_1_spis	0xFFDA3000	0xFFDA303C

Offset: 0x3C

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															rxoicr RO 0x0

rxoicr Fields

Bit	Name	Description	Access	Reset
0	rxoicr	Clear Receive FIFO Overflow Interrupt. This register reflects the status of the interrupt. A read from this register clears the ssi_rxo_intr interrupt; writing has no effect.	RO	0x0

rxuicr

Receive FIFO Underflow Interrupt Clear Register

Module Instance	Base Address	Register Address
i_spis_0_spis	0xFFDA2000	0xFFDA2040
i_spis_1_spis	0xFFDA3000	0xFFDA3040

Offset: 0x40

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															rxuicr RO 0x0

rxuicr Fields

Bit	Name	Description	Access	Reset
0	rxuicr	Clear Receive FIFO Underflow Interrupt. This register reflects the status of the interrupt. A read from this register clears the ssi_rxu_intr interrupt; writing has no effect.	RO	0x0

icr

Interrupt Clear Register

Module Instance	Base Address	Register Address
i_spis_0_spis	0xFFDA2000	0xFFDA2048
i_spis_1_spis	0xFFDA3000	0xFFDA3048

Offset: 0x48

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															icr RO 0x0

icr Fields

Bit	Name	Description	Access	Reset
0	icr	Clear Interrupts. This register is set if any of the interrupts below are active. A read clears the ssi_txo_intr, ssi_rxu_intr, ssi_rxo_intr, and the ssi_mst_intr interrupts. Writing to this register has no effect.	RO	0x0

dmacr

DMA Control Register.
This register is only valid when DW_apb_ssi is configured with a set of
DMA Controller interface signals (SSI_HAS_DMA = 1). When DW_apb_ssi is not configured for DMA operation, this register will not exist and writing
to the register's address will have no effect; reading from this register
address will return zero. The register is used to enable the DMA Controller interface operation.

Module Instance	Base Address	Register Address
i_spis_0_spis	0xFFDA2000	0xFFDA204C
i_spis_1_spis	0xFFDA3000	0xFFDA304C

Offset: 0x4C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														tdmae RW 0x0	rdmae RW 0x0

dmacr Fields

Bit	Name	Description	Access	Reset
1	tdmae	Transmit DMA Enable. This bit enables/disables the transmit FIFO DMA channel. 0 = Transmit DMA disabled 1 = Transmit DMA enabled **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0
0	rdmae	Receive DMA Enable. This bit enables/disables the receive FIFO DMA channel 0 = Receive DMA disabled 1 = Receive DMA enabled **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0

dmatdlr

```
        DMA Transmit Data Level.
This register is only valid when the DW_apb_ssi is configured with a
set
of DMA interface signals (SSI_HAS_DMA = 1). When DW_apb_ssi is not
configured for DMA operation, this register will not exist and writing
to its address will have no effect; reading from its address will
return zero.
```

Module Instance	Base Address	Register Address
i_spis_0_spis	0xFFDA2000	0xFFDA2050
i_spis_1_spis	0xFFDA3000	0xFFDA3050

Offset: 0x50

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								dmatdl RW 0x0							

dmatdlr Fields

Bit	Name	Description	Access	Reset
7:0	dmatdl	Transmit Data Level. This bit field controls the level at which a DMA request is made by the transmit logic. It is equal to the watermark level; that is, the dma_tx_req signal is generated when the number of valid data entries in the transmit FIFO is equal to or below this field value, and TDMAE = 1.	RW	0x0

dmardlr

DMA Receive Data Level.
This register is only valid when DW_apb_ssi is configured with a set of
DMA interface signals (SSI_HAS_DMA = 1). When DW_apb_ssi is not configured
for DMA operation, this register will not exist and writing to its address
will have no effect; reading from its address will return zero.

Module Instance	Base Address	Register Address
i_spis_0_spis	0xFFDA2000	0xFFDA2054
i_spis_1_spis	0xFFDA3000	0xFFDA3054

Offset: 0x54

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								dmardl RW 0x0							

dmardlr Fields

Bit	Name	Description	Access	Reset
7:0	dmardl	Receive Data Level. This bit field controls the level at which a DMA request is made by the receive logic. The watermark level = DMARDL+1; that is, dma_rx_req is generated when the number of valid data entries in the receive FIFO is equal to or above this field value + 1, and RDMAE=1.	RW	0x0

idr

Identification Register.
This register contains the peripherals identification code, which is written into the register at configuration time using coreConsultant.

Module Instance	Base Address	Register Address
i_spis_0_spis	0xFFDA2000	0xFFDA2058
i_spis_1_spis	0xFFDA3000	0xFFDA3058

Offset: 0x58

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
idr RO 0x5510005															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
idr RO 0x5510005															

idr Fields

Bit	Name	Description	Access	Reset
31:0	idr	This filed contains the peripherals identification code, 0x05510005.	RO	0x5510005

spi_version_id

coreKit Version ID Register

Module Instance	Base Address	Register Address
i_spis_0_spis	0xFFDA2000	0xFFDA205C
i_spis_1_spis	0xFFDA3000	0xFFDA305C

Offset: 0x5C

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
spi_version_id RW 0x3332322A															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
spi_version_id RW 0x3332322A															

spi_version_id Fields

Bit	Name	Description	Access	Reset
31:0	spi_version_id	Contains the hex representation of the Synopsys component version. Consists of ASCII value for each number in the version.	RW	0x3332322A

dr

The data register is a 16-bit read/write buffer for the transmit/receive FIFOs. When the register is read, data in the receive FIFO buffer is accessed. When it is written to, data are moved into the transmit FIFO buffer; a write can occur only when SSI_EN = 1. FIFOs are reset when SSI_EN = 0.Note that the data register occupies thirty-six 32-bit address locations of the memory map to facilitate burst transfers. Writing to any of these address locations has the same effect as pushing the data from the bus into the TxFIFO and reading from any of these location has the same effect as popping data from the RxFIFO onto the bus.

Module Instance	Base Address	Register Address
i_spis_0_spis	0xFFDA2000	0xFFDA2060 to 0xFFDA20EC
i_spis_1_spis	0xFFDA3000	0xFFDA3060 to 0xFFDA30EC

Offset: 0x60 to 0xEC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
dr RW 0x0															

dr Fields

Bit	Name	Description	Access	Reset
15:0	dr	Data Register. When writing to this register, you must right-justify the data. Read data are automatically right-justified. Read = Receive FIFO buffer Write = Transmit FIFO buffer	RW	0x0

msticr

Multi-Master Interrupt Clear Register

Module Instance	Base Address	Register Address
i_spis_0_spis	0xFFDA2000	0xFFDA2044
i_spis_1_spis	0xFFDA3000	0xFFDA3044

Offset: 0x44

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															msticr RO 0x0

msticr Fields

Bit	Name	Description	Access	Reset
0	msticr	Clear Multi-Master Contention Interrupt. This register reflects the status of the interrupt. A read from this register clears the ssi_mst_intr interrupt; writing has no effect.	RO	0x0

Document Revision History

Table 19-7: Document Revision History

Date	Version	Changes
October 2016	2016.10.28	Maintenance release.
May 2016	2016.05.27	Maintenance release.
May 2016	2016.05.03	Maintenance release.
November 2015	2015.11.02	• Renamed "Interface Pins" section to "Interface to HPS I/O" and moved it under the "SPI Controller Signal Description" section • Moved "FPGA Routing" section under "SPI Controller Signal Description" Section • Added Clock Gating information • Added Multi-Master mode to "Features of the SPI Controller" section • Updated "RXD Sample Delay" section • Updated "Glue Logic for Master Port ss_in_n" section
May 2015	2015.05.04	Maintenance release.
December 2014	2014.12.15	• Maintenance release. • Added *Taking the SPI Out of Reset* section.
August 2014	2014.08.18	Initial release.

I²C Controller 20

2016.10.28

a10_5v4

The I²C controller provides support for a communication link between integrated circuits on a board. It is a simple two-wire bus which consists of a serial data line (SDA) and a serial clock (SCL) for use in applications such as temperature sensors and voltage level translators to EEPROMs, A/D and D/A converters, CODECs, and many types of microprocessors. †

The hard processor system (HPS) provides five I²C controllers to enable system software to communicate serially with I²C buses. Each I²C controller can operate in master or slave mode, and support standard mode of up to 100 Kbps or fast mode of up to 400 Kbps. These I²C controllers are instances of the Synopsys® DesignWare® APB I²C (DW_apb_i2c) controller.

Each I²C controller must be programmed to operate in either master or slave mode only. Operating as a master and slave simultaneously is not supported. † (53)

Features of the I²C Controller

The I²C controller has the following features:

- Maximum clock speed of up to 400 Kbps
- Standard clock speed 100 kbps
- One of the following I²C operations:
  - A master in an I²C system and programmed only as a master †
  - A slave in an I²C system and programmed only as a slave †
- 7- or 10-bit addressing †
- Mixed read and write combined-format transactions in both 7-bit and 10-bit addressing mode †
- Bulk transmit mode †

(53) Portions © 2016 Synopsys, Inc. Used with permission. All rights reserved. Synopsys & DesignWare are registered trademarks of Synopsys, Inc. All documentation is provided "as is" and without any warranty. Synopsys expressly disclaims any and all warranties, express, implied, or otherwise, including the implied warranties of merchantability, fitness for a particular purpose, and non-infringement, and any warranties arising out of a course of dealing or usage of trade.

†Paragraphs marked with the dagger (†) symbol are Synopsys Proprietary. Used with permission.

- Transmit and receive buffers †
- Handles bit and byte waiting at all bus speeds †
- DMA handshaking interface †

Three of the five I²Cs, provide support for EMAC communication. They provide flexibility for the EMACs to use MDIO or I²C for PHY communication and can also be used as general purpose.

- `I2C_EMAC0`
- `I2C_EMAC1`
- `I2C_EMAC2`

The remaining two I²Cs are intended for general purpose.

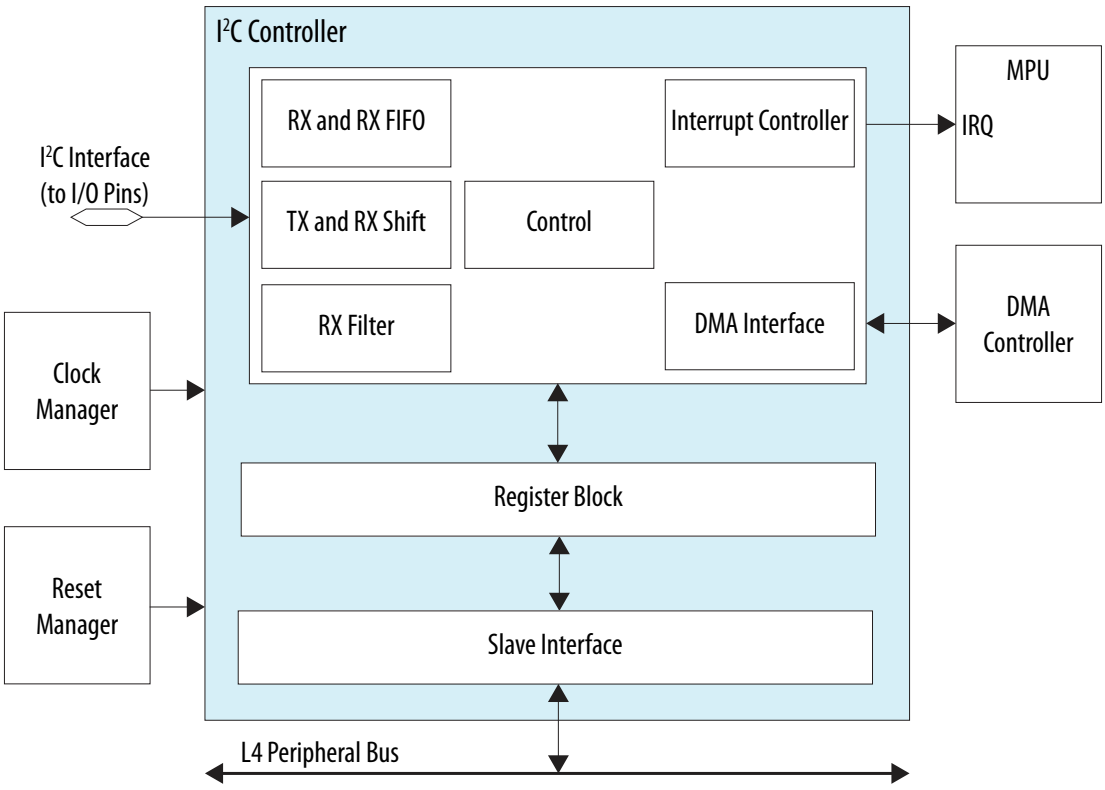
- `I2C0`
- `I2C1`

I²C Controller Block Diagram and System Integration

The I²C controller consists of an internal slave interface, an I²C interface, and FIFO logic to buffer data between the two interfaces. †

The host processor accesses data, control, and status information about the I²C controller through a 32-bit slave interface.

Figure 20-1: I²C Controller Block Diagram



The I²C controller consists of the following modules and interfaces:

- Slave interface for control and status register (CSR) accesses and DMA transfers, allowing a master to access the CSRs and the DMA to read or write data directly.
- Two FIFO buffers for transmit and receive data, which hold the Rx FIFO and Tx FIFO buffer register banks and controllers, along with their status levels. †
- Shift logic for parallel-to-serial and serial-to-parallel conversion
  - Rx shift – Receives data into the design and extracts it in byte format. †
  - Tx shift – Presents data supplied by CPU for transfer on the I²C bus. †
- Control logic responsible for implementing the I²C protocol.
- DMA interface that generates handshaking signals to the DMA controller in order to automate the data transfer without CPU intervention. †
- Interrupt controller that generates raw interrupts and interrupt flags, allowing them to be set and cleared. †
- Receive filter for detecting events, such as start and stop conditions, in the bus; for example, start, stop and arbitration lost. †

I²C Controller Signal Description

All instances of the I²C controller connect to external pins through pin multiplexers. Pin multiplexing allows all instances to function simultaneously and independently. The pins must be connected to a pull-up resistors and the I²C bus capacitance cannot exceed 400 pF.

There are five instances of the I²C which can be routed to the HPS I/O pins. Three of these I²C modules can be used for PHY management by the three Ethernet Media Access Controllers within the HPS. For more information about routing the I²C signals to the FPGA and HPS I/O pins, refer to the *HPS Component Interfaces* chapter.

Table 20-1: I²C Controller Interface HPS I/O Pins

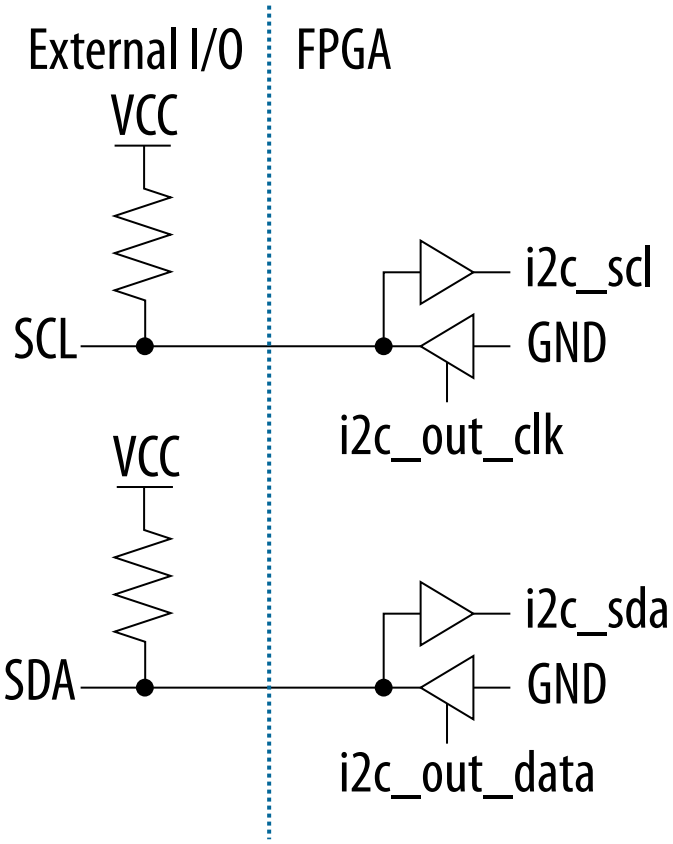
Pin Name	Signal Width	Direction	Description
SCL	1 bit	Bidirectional	Serial clock
SDA	1 bit	Bidirectional	Serial data

Table 20-2: HPS I²C Signals for FPGA Routing

Signal Name	Signal Width	Direction	Description
i2c<#>_scl	1 bit	Input	Incoming I²C clock source. This is the input SCL signal

Signal Name	Signal Width	Direction	Description
i2c<#>_out_clk	1 bit	Output	Outgoing I²C clock enable. Output SCL signal. This signal is logically inverted and is synchronous to the HPS peripheral clock
i2c<#>_sda	1 bit	Input	Incoming I²C data. This is the input SDA signal.
i2c<#>_out_data	1 bit	Output	Outgoing I²C data enable. Output SDA signal. This signal is logically inverted and is synchronous to the HPS peripheral clock.

Figure 20-2: I²C Interface in FPGA Fabric



The figure above shows the typical connection on the I²C interface in FPGA fabric with `alt_iobuff`.

For both I2C clock and data, external IO pins are open drain connection. When output enables `i2c_out_data` and `i2c_out_clk` are asserted, external signal will be driven to ground.

Related Information

HPS Component Interfaces on page 28-13
For more information on how the I2C Signals are routed to the FPGA and HPS I/O pins, refer to this chapter.

Functional Description of the I2C Controller

Feature Usage

The I2C controller can operate in standard mode (with data rates of up to 100 Kbps) or fast mode (with data rates less than or equal to 400 Kbps). Additionally, fast mode devices are downward compatible. For instance, fast mode devices can communicate with standard mode devices in 0 to 100 Kbps I2C bus system. However, standard mode devices are not upward compatible and should not be incorporated in a fast-mode I2C bus system as they cannot follow the higher transfer rate and therefore unpredictable states would occur. †

You can attach any I2C controller to an I2C-bus and every device can talk with any master, passing information back and forth. There needs to be at least one master (such as a microcontroller or DSP) on the bus and there can be multiple masters, which require them to arbitrate for ownership. Multiple masters and arbitration are explained later in this chapter. †

Behavior

You can control the I2C controller via software to be in either mode:

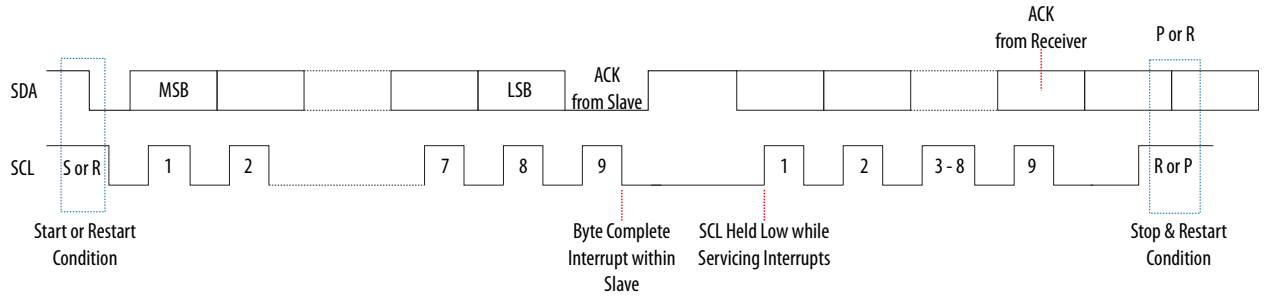
- An I2C master only, communicating with other I2C slaves.
- An I2C slave only, communicating with one or more I2C masters.

The master is responsible for generating the clock and controlling the transfer of data. The slave is responsible for either transmitting or receiving data to/from the master. The acknowledgement of data is sent by the device that is receiving data, which can be either a master or a slave. As mentioned previously, the I2C protocol also allows multiple masters to reside on the I2C bus and uses an arbitration procedure to determine bus ownership. †

Each slave has a unique address that is determined by the system designer. When a master wants to communicate with a slave, the master transmits a START/RESTART condition that is then followed by the slave's address and a control bit (R/W) to determine if the master wants to transmit data or receive data from the slave. The slave then sends an acknowledge (ACK) pulse after the address. †

If the master (master-transmitter) is writing to the slave (slave-receiver), the receiver receives one byte of data. This transaction continues until the master terminates the transmission with a STOP condition. If the master is reading from a slave (master-receiver), the slave transmits (slave-transmitter) a byte of data to the master, and the master then acknowledges the transaction with an ACK pulse. This transaction continues until the master terminates the transmission by not acknowledging (NACK) the transaction after the last byte is received, and then the master issues a STOP condition or addresses another slave after issuing a RESTART condition. †

Figure 20-3: Data Transfer on the I²C Bus †



The I²C controller is a synchronous serial interface. The SDA line is a bidirectional signal and changes only while the SCL line is low, except for STOP, START, and RESTART conditions. The output drivers are open-drain or open-collector to perform wire-AND functions on the bus. The maximum number of devices on the bus is limited by only the maximum capacitance specification of 400 pF. Data is transmitted in byte packages. †

START and STOP Generation

When operating as a master, putting data into the transmit FIFO causes the I²C controller to generate a START condition on the I²C bus. In order for the master to complete the transfer and issue a STOP condition it must find a transmit FIFO entry tagged with a stop bit. Allowing the transmit FIFO to empty without a stop bit set, the master will stall the transfer by holding the SCL line low. †

When operating as a slave, the I²C controller does not generate START and STOP conditions, as per the protocol. However, if a read request is made to the I²C controller, it holds the SCL line low until read data has been supplied to it. This stalls the I²C bus until read data is provided to the slave I²C controller, or the I²C controller slave is disabled by writing a 0 to bit 0 of `IC_ENABLE` register. †

Combined Formats

The I²C controller supports mixed read and write combined format transactions in both 7-bit and 10-bit addressing modes. †

The I²C controller does not support mixed address and mixed address format—that is, a 7-bit address transaction followed by a 10-bit address transaction or vice versa—combined format transactions. †

To initiate combined format transfers, the `IC_RESTART_EN` bit in the `IC_CON` register should be set to 1. With this value set and operating as a master, when the I²C controller completes an I²C transfer, it checks the transmit FIFO and executes the next transfer. If the direction of this transfer differs from the previous transfer, the combined format is used to issue the transfer. If the `IC_RESTART_EN` is 0, a STOP will be issued followed by a START condition. Another way to generate the RESTART condition is to set the Restart bit [10] of the `DATA_CMD` register. Regardless if the direction of the transfer changes or not the RESTART condition will be generated.†

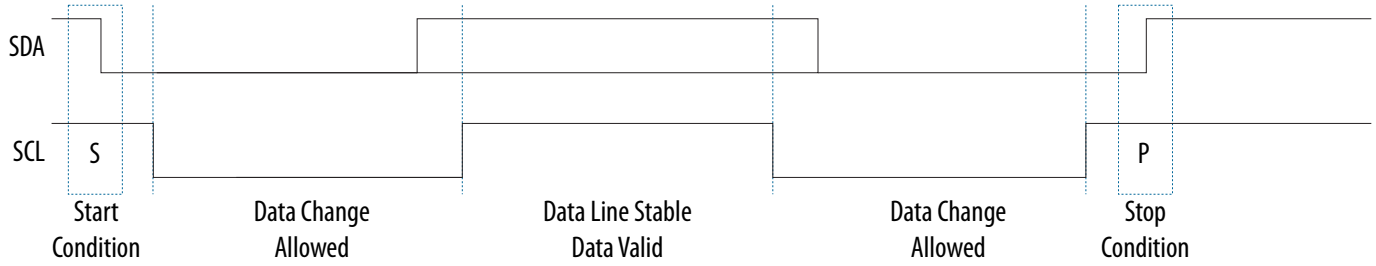
Protocol Details

START and STOP Conditions

When the bus is idle, both the SCL and SDA signals are pulled high through pull-up resistors on the bus. When the master wants to start a transmission on the bus, the master issues a START condition. This is defined to be a high-to-low transition of the SDA signal while SCL is 1. When the master wants to terminate the transmission, the master issues a STOP condition. This is defined to be a low-to-high transition of the SDA line while SCL is 1. †

The following figure shows the timing of the START and STOP conditions. When data is being transmitted on the bus, the SDA line must be stable when SCL is 1. †

Figure 20-4: Timing Diagram for START and STOP Conditions



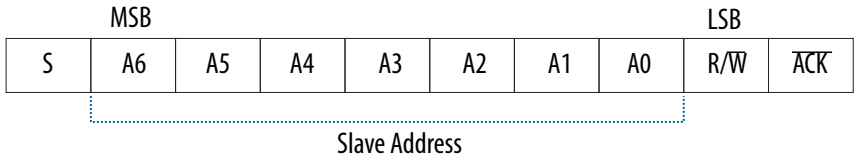
The signal transitions for the START or STOP condition, as shown in the figure, reflect those observed at the output signals of the master driving the I^2C bus. Care should be taken when observing the SDA or SCL signals at the input signals of the slave(s), because unequal line delays may result in an incorrect SDA or SCL timing relationship. †

Addressing Slave Protocol

7-Bit Address Format

During the 7-bit address format, the first seven bits (bits 7:1) of the first byte set the slave address and the LSB bit (bit 0) is the R/W bit as shown in the following figure. When bit 0 (R/W) is set to 0, the master writes to the slave. When bit 0 (R/W) is set to 1, the master reads from the slave. †

Figure 20-5: 7- Bit Address Format

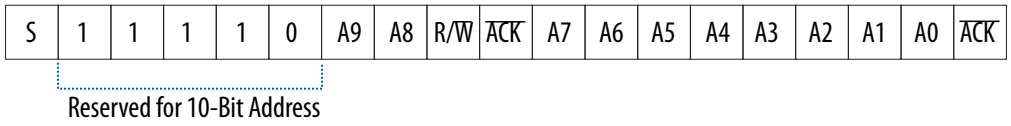


S: Start Condition
R/W: Read/Write Pulse
ACK: Acknowledge (Sent by Slave)

10-Bit Address Format

During 10-bit addressing, two bytes are transferred to set the 10-bit address. The transfer of the first byte contains the following bit definition. The first five bits (bits 7:3) notify the slaves that this is a 10-bit transfer followed by the next two bits (bits 2:1), which set the slaves address bits 9:8, and the LSB bit (bit 0) is the R/W bit. The second byte transferred sets bits 7:0 of the slave address. †

Figure 20-6: 10-Bit Address Format



S: Start Condition
R/W: Read/Write Pulse
ACK: Acknowledge (Sent by Slave)

The following table defines the special purpose and reserved first byte addresses. †

Table 20-3: I²C Definition of Bits in First Byte

Slave Address	R/W Bit	Description
0000 000	0	General call address. The I²C controller places the data in the receive buffer and issues a general call interrupt.
0000 000	1	START byte. For more details, refer to "START BYTE Transfer Protocol"
0000 001	X	CBUS address. The I²C controller ignores these accesses.
0000 010	X	Reserved
0000 011	X	Reserved
0000 1XX	X	Unused
1111 1XX	X	Reserved
1111 0XX	X	10-bit slave addressing.

Note to Table: 'X' indicates do not care.

Related Information

START BYTE Transfer Protocol on page 20-10

Transmitting and Receiving Protocol

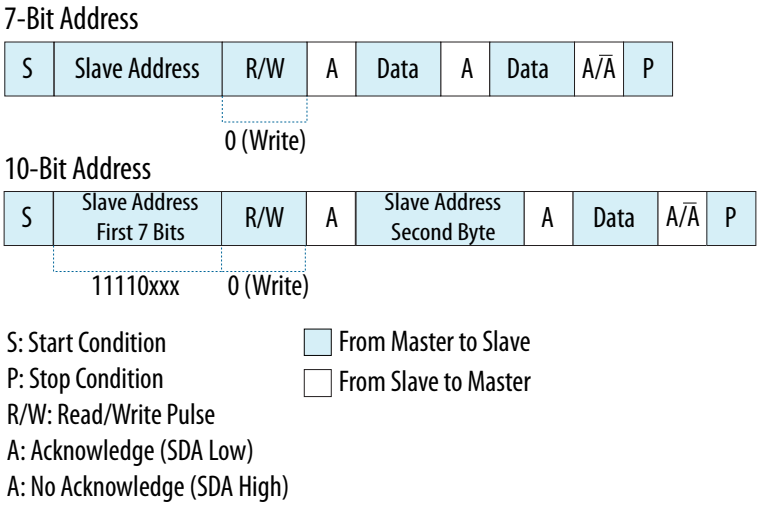
The master can initiate data transmission and reception to or from the bus, acting as either a master-transmitter or master-receiver. A slave responds to requests from the master to either transmit data or receive data to or from the bus, acting as either a slave-transmitter or slave-receiver, respectively. †

Master-Transmitter and Slave-Receiver

All data is transmitted in byte format, with no limit on the number of bytes transferred per data transfer. After the master sends the address and R/W bit or the master transmits a byte of data to the slave, the slave-receiver must respond with the acknowledge signal (ACK). When a slave-receiver does not respond with an ACK pulse, the master aborts the transfer by issuing a STOP condition. The slave must leave the SDA line high so that the master can abort the transfer. †

If the master-transmitter is transmitting data as shown in the following figure, then the slave-receiver responds to the master-transmitter with an ACK pulse after every byte of data is received. †

Figure 20-7: Master-Transmitter Protocol †



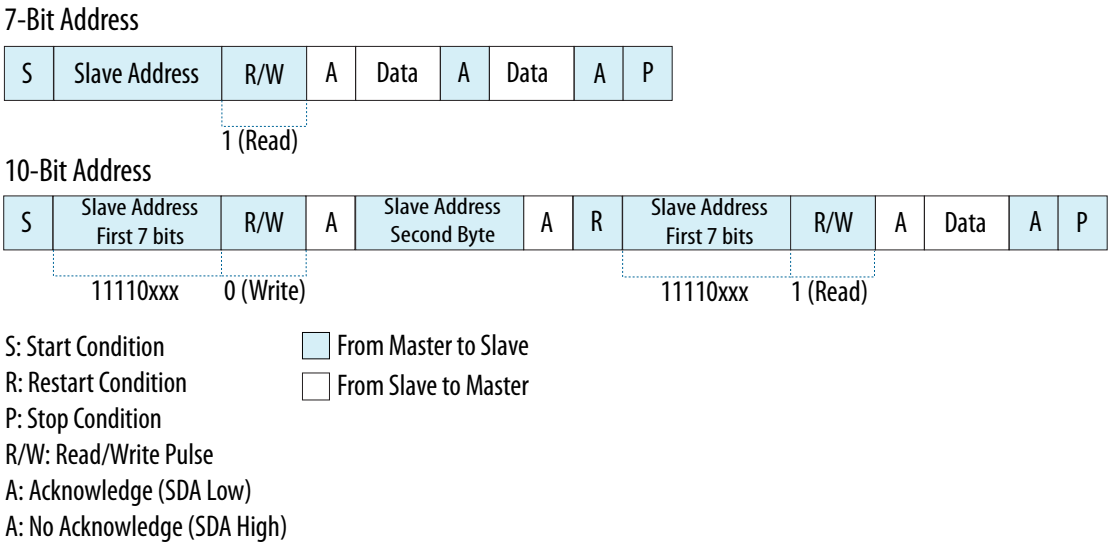
Master-Receiver and Slave-Transmitter

If the master is receiving data as shown in the following figure, then the master responds to the slave-transmitter with an ACK pulse after a byte of data has been received, except for the last byte. This is the way the master-receiver notifies the slave-transmitter that this is the last byte. The slave-transmitter relinquishes the SDA line after detecting the No Acknowledge (NACK) bit so that the master can issue a STOP condition. †

When a master does not want to relinquish the bus with a STOP condition, the master can issue a RESTART condition. This is identical to a START condition except it occurs after the ACK pulse. Operating in master mode, the I^2C controller can then communicate with the same slave using a transfer of a different direction. For a description of the combined format transactions that the I^2C controller supports, refer to "Combined Formats" section of this chapter. †

Note: The I^2C controller must be inactive on the serial port before the target slave address register, `IC_TAR`, can be reprogrammed. †

Figure 20-8: Master-Receiver Protocol †



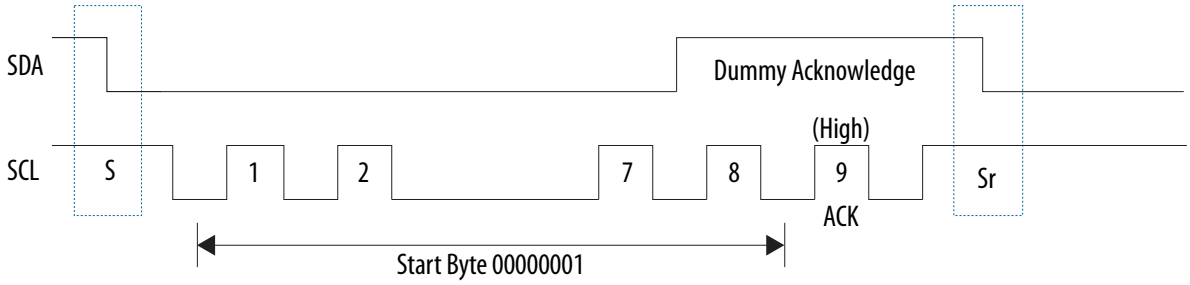
Related Information

Combined Formats on page 20-6

START BYTE Transfer Protocol

The START BYTE transfer protocol is set up for systems that do not have an on-board dedicated I^2C hardware module. When the I^2C controller is set as a slave, it always samples the I^2C bus at the highest speed supported so that it never requires a START BYTE transfer. However, when I^2C controller is set as a master, it supports the generation of START BYTE transfers at the beginning of every transfer in case a slave device requires it. This protocol consists of seven zeros being transmitted followed by a 1, as illustrated in the following figure. This allows the processor that is polling the bus to under-sample the address phase until the microcontroller detects a 0. Once the microcontroller detects a 0, it switches from the under sampling rate to the correct rate of the master. †

Figure 20-9: START BYTE Transfer †



The START BYTE has the following procedure: †

1. Master generates a START condition. †
2. Master transmits the START byte (0000 0001). †
3. Master transmits the ACK clock pulse. (Present only to conform with the byte handling format used on the bus) †
4. No slave sets the ACK signal to 0. †
5. Master generates a RESTART (R) condition. †

A hardware receiver does not respond to the START BYTE because it is a reserved address and resets after the RESTART condition is generated. †

Multiple Master Arbitration

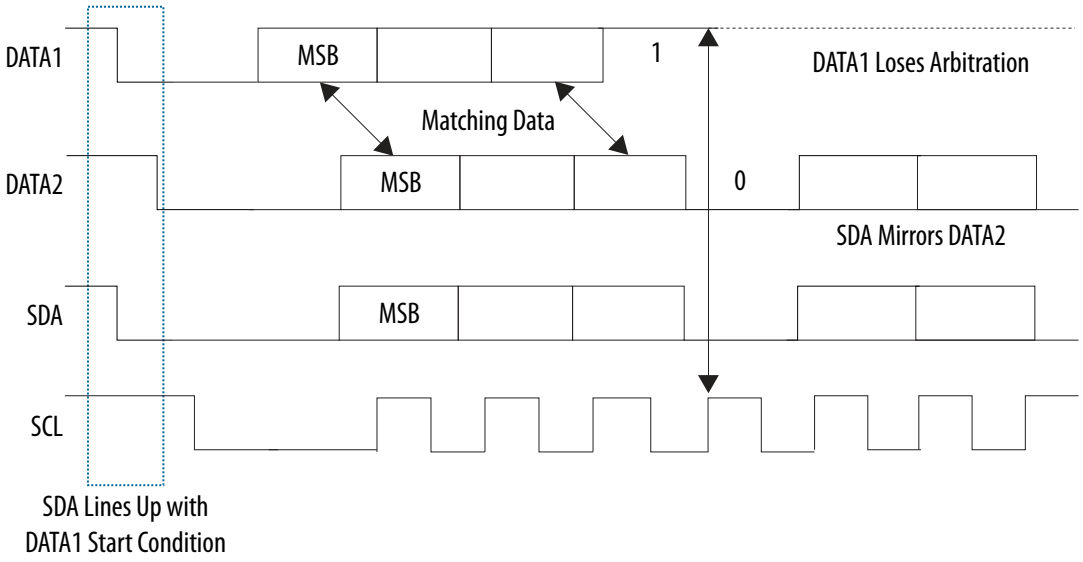
The I^2C controller bus protocol allows multiple masters to reside on the same bus. If there are two masters on the same I^2C-bus, there is an arbitration procedure if both try to take control of the bus at the same time by simultaneously generating a START condition. Once a master (for example, a microcontroller) has control of the bus, no other master can take control until the first master sends a STOP condition and places the bus in an idle state. †

Arbitration takes place on the SDA line, while the SCL line is 1. The master, which transmits a 1 while the other master transmits 0, loses arbitration and turns off its data output stage. The master that lost arbitration can continue to generate clocks until the end of the byte transfer. If both masters are addressing the same slave device, the arbitration could go into the data phase. †

Upon detecting that it has lost arbitration to another master, the I^2C controller stops generating SCL. †

The following figure illustrates the timing of two masters arbitrating on the bus.

Figure 20-10: Multiple Master Arbitration †



The bus control is determined by address or master code and data sent by competing masters, so there is no central master nor any order of priority on the bus. †

Arbitration is not allowed between the following conditions: †

- A RESTART condition and a data bit †
- A STOP condition and a data bit †
- A RESTART condition and a STOP condition †

Slaves are not involved in the arbitration process. †

Clock Synchronization

When two or more masters try to transfer information on the bus at the same time, they must arbitrate and synchronize the SCL clock. All masters generate their own clock to transfer messages. Data is valid only during the high period of SCL clock. Clock synchronization is performed using the wired-AND connection to the SCL signal. When the master transitions the SCL clock to 0, the master starts counting the low time of the SCL clock and transitions the SCL clock signal to 1 at the beginning of the next clock period. However, if another master is holding the SCL line to 0, then the master goes into a HIGH wait state until the SCL clock line transitions to 1. †

All masters then count off their high time, and the master with the shortest high time transitions the SCL line to 0. The masters then counts out their low time and the one with the longest low time forces the other master into a HIGH wait state. Therefore, a synchronized SCL clock is generated, which is illustrated in the following figure. Optionally, slaves may hold the SCL line low to slow down the timing on the I^2C bus. †

Figure 20-11: Multiple Master Clock Synchronization †

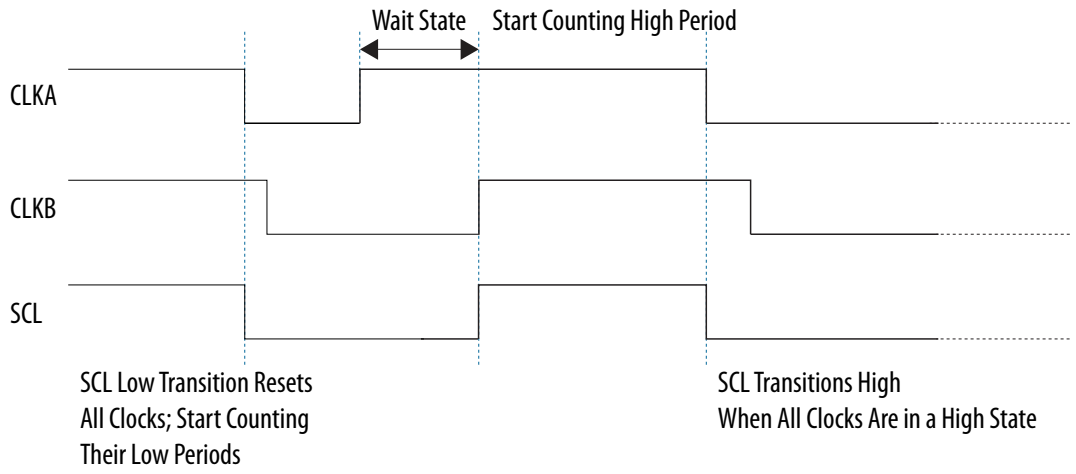
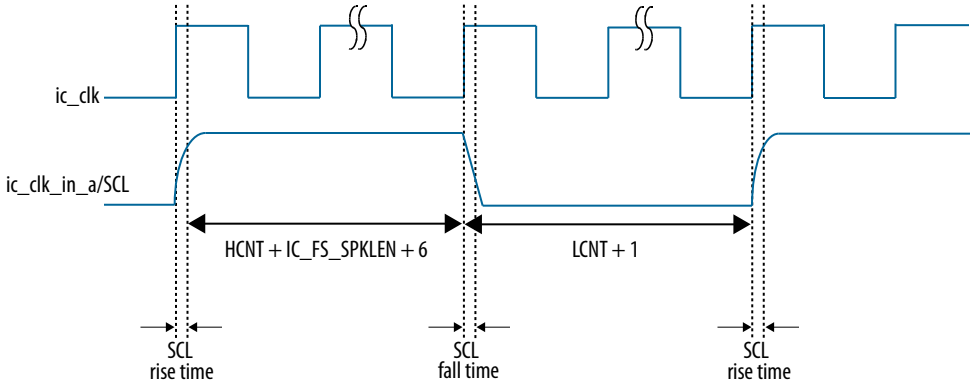


Figure 20-12: Impact of SCL Rise Time and Fall Time on Generated SCL



The following equations can be used to compute SCL high and low time:

```
SCL_High_time = [(HCNT + IC_FS_SPKLEN + 6)*ic_clk] + SCL_Fall_time
SCL_Low_time = [(LCNT + 1)*ic_clk] - SCL_Fall_time + SCL_Rise_time
```

Clock Frequency Configuration

When you configure the I²C controller as a master, the SCL count registers must be set before any I²C bus transaction can take place in order to ensure proper I/O timing. † There are four SCL count registers:

- Standard speed I²C clock SCL high count, `IC_SS_SCL_HCNT` †
- Standard speed I²C clock SCL low count, `IC_SS_SCL_LCNT` †
- Fast speed I²C clock SCL high count, `IC_FS_SCL_HCNT` †
- Fast speed I²C clock SCL low count, `IC_FS_SCL_LCNT` †

It is not necessary to program any of the SCL count registers if the I²C controller is enabled to operate only as an I²C slave, since these registers are used only to determine the SCL timing requirements for operation as an I²C master. †

Minimum High and Low Counts

When the I²C controller operates as an I²C master in both transmit and receive transfers, the minimum value that can be programmed in the SCL low count registers is 8 while the minimum value allowed for the SCL high count registers is 6. †

The minimum value of 8 for the low count registers is due to the time required for the I²C controller to drive SDA after a negative edge of SCL. The minimum value of 6 for the high count register is due to the time required for the I²C controller to sample SDA during the high period of SCL. †

The I²C controller adds one cycle to the low count register values in order to generate the low period of the SCL clock.

The I²C controller adds seven cycles to the high count register values in order to generate the high period of the SCL clock. This is due to the following factors: †

- The digital filtering applied to the SCL line incurs a delay of four `l4_sp_clk` cycles. This filtering includes metastability removal and a 2-out-of-3 majority vote processing on SDA and SCL edges. †
- Whenever SCL is driven 1 to 0 by the I^2C controller—that is, completing the SCL high time—an internal logic latency of three `l4_sp_clk` cycles incurs. †

Consequently, the minimum SCL low time of which the I^2C controller is capable is nine (9) `l4_sp_clk` periods (8+1), while the minimum SCL high time is thirteen (13) `l4_sp_clk` periods (6+1+3+3). †

Note: The `ic_fs_spklen` register must be set before any I^2C bus transaction can take place to ensure stable operation. This register sets the duration measured in `ic_clk` cycles, of the longest spike in the SCL or SDA lines that will be filtered out by the spike suppression logic.†

Calculating High and Low Counts

The calculations below show an example of how to calculate SCL high and low counts for each speed mode in the I^2C controller.

The equation to calculate the proper number of `l4_sp_clk` clock pulses required for setting the proper SCL clocks high and low times is as follows: †

Table 20-4: Equation

IC_HCNT = ceil(MIN_SCL_`HIGHtime`*OSCFREQ)

IC_LCNT = ceil(MIN_SCL_`LOWtime`*OSCFREQ)

MIN_SCL_HIGHtime = minimum high period

MIN_SCL_HIGHtime =

4000 ns for 100 kbps

600 ns for 400 kbps

60 ns for 3.4 Mbs, bus loading = 100pF

160 ns for 3.4 Mbs, bus loading = 400pF

MIN_SCL_LOWtime = minimum low period

MIN_SCL_LOWtime =

4700 ns for 100 kbps

1300 ns for 400 kbps

120 ns for 3.4Mbs, bus loading = 100pF

320 ns for 3.4Mbs, bus loading = 400pF

OSCFREQ = `l4_sp_clk` clock frequency (Hz)

Example 20-1: Calculating High and Low Counts

```
OSCFREQ = 100 MHz
I2Cmode = fast, 400 kbps
```

```
MIN_SCL_HIGHtime = 600 ns
MIN_SCL_LOWtime = 1300 ns

IC_HCNT = ceil(600 ns * 100 MHz) IC_HCNTSCL PERIOD = 60
IC_LCNT = ceil(1300 ns * 100 MHz) IC_LCNTSCL PERIOD = 130
Actual MIN_SCL_HIGHtime = 60*(1/100 MHz) = 600 ns
Actual MIN_SCL_LOWtime = 130*(1/100 MHz) = 1300 ns †
```

SDA Hold Time

The I²C protocol specification requires 300 ns of hold time on the SDA signal in standard and fast speed modes. Board delays on the SCL and SDA signals can mean that the hold time requirement is met at the I²C master, but not at the I²C slave (or vice-versa). As each application encounters differing board delays, the I²C controller contains a software programmable register, `IC_SDA_HOLD`, to enable dynamic adjustment of the SDA hold time. `IC_SDA_HOLD` effects both slave-transmitter and master mode.

DMA Controller Interface

The I²C controller supports DMA signaling to indicate when data is ready to be read or when the transmit FIFO needs data. This support requires 2 DMA channels, one for transmit data and one for receive data. The I²C controller supports both single and burst DMA transfers. System software can choose the DMA burst mode by programming an appropriate value into the threshold registers. The recommended setting of the FIFO threshold register value is half full.

To enable the DMA controller interface on the I²C controller, you must write to the DMA control register (`DMACR`) bits. Writing a 1 into the `TDMAE` bit field of `DMACR` register enables the I²C controller transmit handshaking interface. Writing a 1 into the `RDMAE` bit field of the `DMACR` register enables the I²C controller receive handshaking interface. †

Related Information

DMA Controller on page 16-1
For details about the DMA burst length microcode setup, refer to the *DMA controller* chapter.

Clocks

Each I²C controller is connected to the `l4_sp_clk` clock, which clocks transfers in standard and fast mode. The clock input is driven by the clock manager.

Related Information

Clock Manager on page 2-1
For more information, refer to *Clock Manager* chapter.

Resets

Each I²C controller has a separate reset signal. The reset manager drives the signals on a cold or warm reset.

Related Information

Reset Manager on page 3-1
For more information, refer to *Reset Manager* chapter.

Taking the I2C Controller Out of Reset

When a cold or warm reset is issued in the HPS, the reset manager resets this module and holds it in reset until software releases it.

After the Cortex-A9 MPCore CPU boots, it can deassert the reset signal by clearing the appropriate bits in the reset manager's corresponding reset register. For details about reset registers, refer to "Module Reset Signals".

Related Information

Modules Requiring Software Deassert on page 3-13

I2C Controller Programming Model

This section describes the programming model for the I2C controllers based on the two master and slave operation modes. †

Note: Each I2C controller should be set to operate only as an I2C master or as an I2C slave, never set both simultaneously. Ensure that bit 6 (`IC_SLAVE_DISABLE`) and 0 (`IC_MASTER_MODE`) of the `IC_CON` register are never set to 0 and 1, respectively. †

Slave Mode Operation

Initial Configuration

To use the I2C controller as a slave, perform the following steps: †

1. Disable the I2C controller by writing a 0 to bit 0 of the `IC_ENABLE` register. †
2. Write to the `IC_SAR` register (bits 9:0) to set the slave address. This is the address to which the I2C controller responds. †

   Note: The reset value for the I2C controller slave address is 0x55. If you are using 0x55 as the slave address, you can safely skip this step.
3. Write to the `IC_CON` register to specify which type of addressing is supported (7- or 10-bit by setting bit 3). Enable the I2C controller in slave-only mode by writing a 0 into bit 6 (`IC_SLAVE_DISABLE`) and a 0 to bit 0 (`MASTER_MODE`). †

   Note: Slaves and masters do not have to be programmed with the same type of addressing 7- or 10-bit address. For instance, a slave can be programmed with 7-bit addressing and a master with 10-bit addressing, and vice versa. †
4. Enable the I2C controller by writing a 1 in bit 0 of the `IC_ENABLE` register. †

   Note: It is recommended that the I2C Slave be brought out of reset only when the I2C bus is `IDLE`. De-asserting the reset when a transfer is ongoing on the bus causes internal flip-flops used to synchronize `SDA` and `SCL` to toggle from a reset value of 1 to the actual value on the bus. In this scenario, if `SDA` toggling from 1 to 0 while `SCL` is 1, thereby causing a false START condition to be detected by the I2C Slave by configuring the I2C with `IC_SLAVE_DISABLE` = 1 and `IC_MASTER_MODE` = 1 so that the Slave interface is disabled after reset. It can then be enabled by programming IC_CON[0] = 0 and IC_CON[6] = 0 after the internal `SDA` and `SCL` have synchronized to the value on the bus; this takes approximately 6 `ic_clk` cycles after reset de-assertion.†

Slave-Transmitter Operation for a Single Byte

When another I²C master device on the bus addresses the I²C controller and requests data, the I²C controller acts as a slave-transmitter and the following steps occur: †

1. The other I²C master device initiates an I²C transfer with an address that matches the slave address in the IC_SAR register of the I²C controller †
2. The I²C controller acknowledges the sent address and recognizes the direction of the transfer to indicate that it is acting as a slave-transmitter. †
3. The I²C controller asserts the RD_REQ interrupt (bit 5 of the IC_RAW_INTR_STAT register) and waits for software to respond. †

   If the RD_REQ interrupt has been masked, due to bit 5 of the IC_INTR_MASK register (M_RD_REQ bit field) being set to 0, then it is recommended that you instruct the CPU to perform periodic reads of the IC_RAW_INTR_STAT register. †

   - Reads that indicate bit 5 of the IC_RAW_INTR_STAT register (R_RD_REQ bit field) being set to 1 must be treated as the equivalent of the RD_REQ interrupt being asserted. †
   - Software must then act to satisfy the I²C transfer. †
   - The timing interval used should be in the order of 10 times the fastest SCL clock period the I²C controller can handle. For example, for 400 Kbps, the timing interval is 25 us. †

     Note: The value of 10 is recommended here because this is approximately the amount of time required for a single byte of data transferred on the I²C bus.†
4. If there is any data remaining in the TX FIFO before receiving the read request, the I²C controller asserts a TX_ABRT interrupt (bit 6 of the IC_RAW_INTR_STAT register) to flush the old data from the TX FIFO. †

   Note: Because the I²C controller's TX FIFO is forced into a flushed/reset state whenever a TX_ABRT event occurs, it is necessary for software to release the I²C controller from this state by reading the IC_CLR_TX_ABRT register before attempting to write into the TX FIFO. For more information, refer to the C_RAW_INTR_STAT register description in the register map.†

   If the TX_ABRT interrupt has been masked, due to of IC_INTR_MASK[6] register (M_TX_ABRT bit field) being set to 0, then it is recommended that the CPU performs periodic reads of the IC_RAW_INTR_STAT register. †

   - Reads that indicate bit 6 (R_TX_ABRT) being set to 1 must be treated as the equivalent of the TX_ABRT interrupt being asserted. †
   - There is no further action required from software. †
   - The timing interval used should be similar to that described in the previous step for the IC_RAW_INTR_STAT[5] register. †
5. Software writes to the DAT bits of the IC_DATA_CMD register with the data to be written and writes a 0 in bit 8.†
6. Software must clear the RD_REQ and TX_ABRT interrupts (bits 5 and 6, respectively) of the IC_RAW_INTR_STAT register before proceeding. †

   If the RD_REQ and/or TX_ABRT interrupts have been masked, then clearing of the IC_RAW_INTR_STAT register will have already been performed when either the R_RD_REQ or R_TX_ABRT bit has been read as 1. †
7. The I²C controller transmits the byte. †
8. The master may hold the I²C bus by issuing a RESTART condition or release the bus by issuing a STOP condition. †

Slave-Receiver Operation for a Single Byte

When another I²C master device on the bus addresses the I²C controller and is sending data, the I²C controller acts as a slave-receiver and the following steps occur:†

1. The other I²C master device initiates an I²C transfer with an address that matches the I²C controller's slave address in the IC_SAR register. †
2. The I²C controller acknowledges the sent address and recognizes the direction of the transfer to indicate that the I²C controller is acting as a slave-receiver. †
3. I²C controller receives the transmitted byte and places it in the receive buffer. †

   Note: If the RX FIFO is completely filled with data when a byte is pushed, then an overflow occurs and the I²C controller continues with subsequent I²C transfers. Because a NACK is not generated, software must recognize the overflow when indicated by the I²C controller (by the R_RX_OVER bit in the IC_INTR_STAT register) and take appropriate actions to recover from lost data. Hence, there is a real time constraint on software to service the RX FIFO before the latter overflow as there is no way to reapply pressure to the remote transmitting master. †
4. I²C controller asserts the RX_FULL interrupt (IC_RAW_INTR_STAT[2] register). †

   If the RX_FULL interrupt has been masked, due to setting IC_INTR_MASK[2] register to 0 or setting IC_TX_TL to a value larger than 0, then it is recommended that the CPU does periodic reads of the IC_STATUS register. Reads of the IC_STATUS register, with bit 3 (RFNE) set at 1, must then be treated by software as the equivalent of the RX_FULL interrupt being asserted. †
5. Software may read the byte from the IC_DATA_CMD register (bits 7:0). †
6. The other master device may hold the I²C bus by issuing a RESTART condition or release the bus by issuing a STOP condition. †

Slave-Transfer Operation for Bulk Transfers

In the standard I²C protocol, all transactions are single byte transactions and the programmer responds to a remote master read request by writing one byte into the slave's TX FIFO. When a slave (slave-transmitter) is issued with a read request (RD_REQ) from the remote master (master-receiver), at a minimum there should be at least one entry placed into the slave-transmitter's TX FIFO. The I²C controller is designed to handle more data in the TX FIFO so that subsequent read requests can receive that data without raising an interrupt to request more data. Ultimately, this eliminates the possibility of significant latencies being incurred between raising the interrupt for data each time had there been a restriction of having only one entry placed in the TX FIFO. †

This mode only occurs when I²C controller is acting as a slave-transmitter. If the remote master acknowledges the data sent by the slave-transmitter and there is no data in the slave's TX FIFO, the I²C controller raises the read request interrupt (RD_REQ) and waits for data to be written into the TX FIFO before it can be sent to the remote master. †

If the RD_REQ interrupt is masked, due to bit 5 (M_RD_REQ) of the IC_INTR_STAT register being set to 0, then it is recommended that the CPU does periodic reads of the IC_RAW_INTR_STAT register. Reads of IC_RAW_INTR_STAT that return bit 5 (R_RD_REQ) set to 1 must be treated as the equivalent of the RD_REQ interrupt referred to in this section.†

The RD_REQ interrupt is raised upon a read request, and like interrupts, must be cleared when exiting the interrupt service handling routine (ISR). The ISR allows you to either write 1 byte or more than 1 byte into the TX FIFO. During the transmission of these bytes to the master, if the master acknowledges the last byte then the slave must raise the RD_REQ again because the master is requesting for more data. †

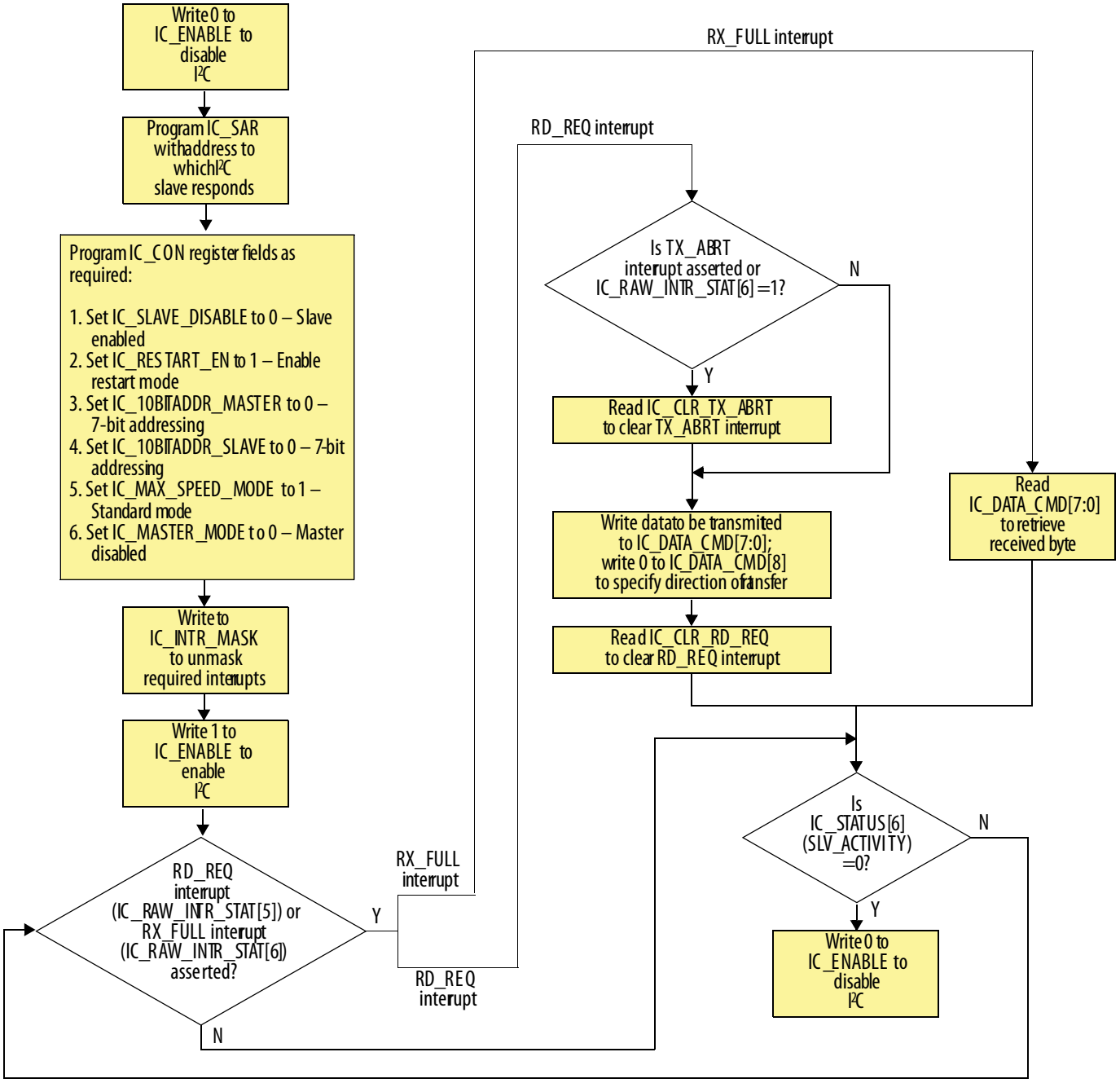
If the programmer knows in advance that the remote master is requesting a packet of n bytes, then when another master addresses the I²C controller and requests data, the TX FIFO could be written with n number bytes and the remote master receives it as a continuous stream of data. For example, the I²C

controller slave continues to send data to the remote master as long as the remote master is acknowledging the data sent and there is data available in the TX FIFO. There is no need to issue RD_REQ again. †

If the remote master is to receive n bytes from the I^2C controller but the programmer wrote a number of bytes larger than n to the TX FIFO, then when the slave finishes sending the requested n bytes, it clears the TX FIFO and ignores any excess bytes. †

The I^2C controller generates a transmit abort (TX_ABRT) event to indicate the clearing of the TX FIFO in this example. At the time an ACK/NACK is expected, if a NACK is received, then the remote master has all the data it wants. At this time, a flag is raised within the slave's state machine to clear the leftover data in the TX FIFO. This flag is transferred to the processor bus clock domain where the FIFO exists and the contents of the TX FIFO are cleared at that time. †

Slave Programming Model



Master Mode Operation

Initial Configuration

For master mode operation, the target address and address format can be changed dynamically without having to disable the I^2C controller. This feature is only applicable when the I^2C controller is acting as a master because the slave requires the component to be disabled before any changes can be made to the address. To use the I^2C controller as a master, perform the following steps: †

For multiple I²C transfers, perform additional writes to the Tx FIFO such that the Tx FIFO does not become empty during the I²C transaction. IF the Tx FIFO is completely emptied at any stage, then the master stalls the transfer by holding the SCL line low because there was no stop bit indicating the master to issue a STOP. The master will complete the transfer when it finds a Tx FIFO entry tagged with a Stop bit.

1. Disable the I²C controller by writing 0 to bit 0 of the `IC_ENABLE` register. †
2. Write to the `IC_CON` register to set the maximum speed mode supported for slave operation (bits 2:1) and to specify whether the I²C controller starts its transfers in 7/10 bit addressing mode when the device is a slave (bit 3). †
3. Write to the `IC_TAR` register the address of the I²C device to be addressed. It also indicates whether a General Call or a START BYTE command is going to be performed by I²C. The desired speed of the I²C controller master-initiated transfers, either 7-bit or 10-bit addressing, is controlled by the `IC_10BITADDR_MASTER` bit field (bit 12). †
4. Enable the I²C controller by writing a 1 in bit 0 of the `IC_ENABLE` register. †
5. Now write the transfer direction and data to be sent to the `IC_DATA_CMD` register. If the `IC_DATA_CMD` register is written before the I²C controller is enabled, the data and commands are lost as the buffers are kept cleared when the I²C controller is not enabled. †

Dynamic IC_TAR or IC_10BITADDR_MASTER Update

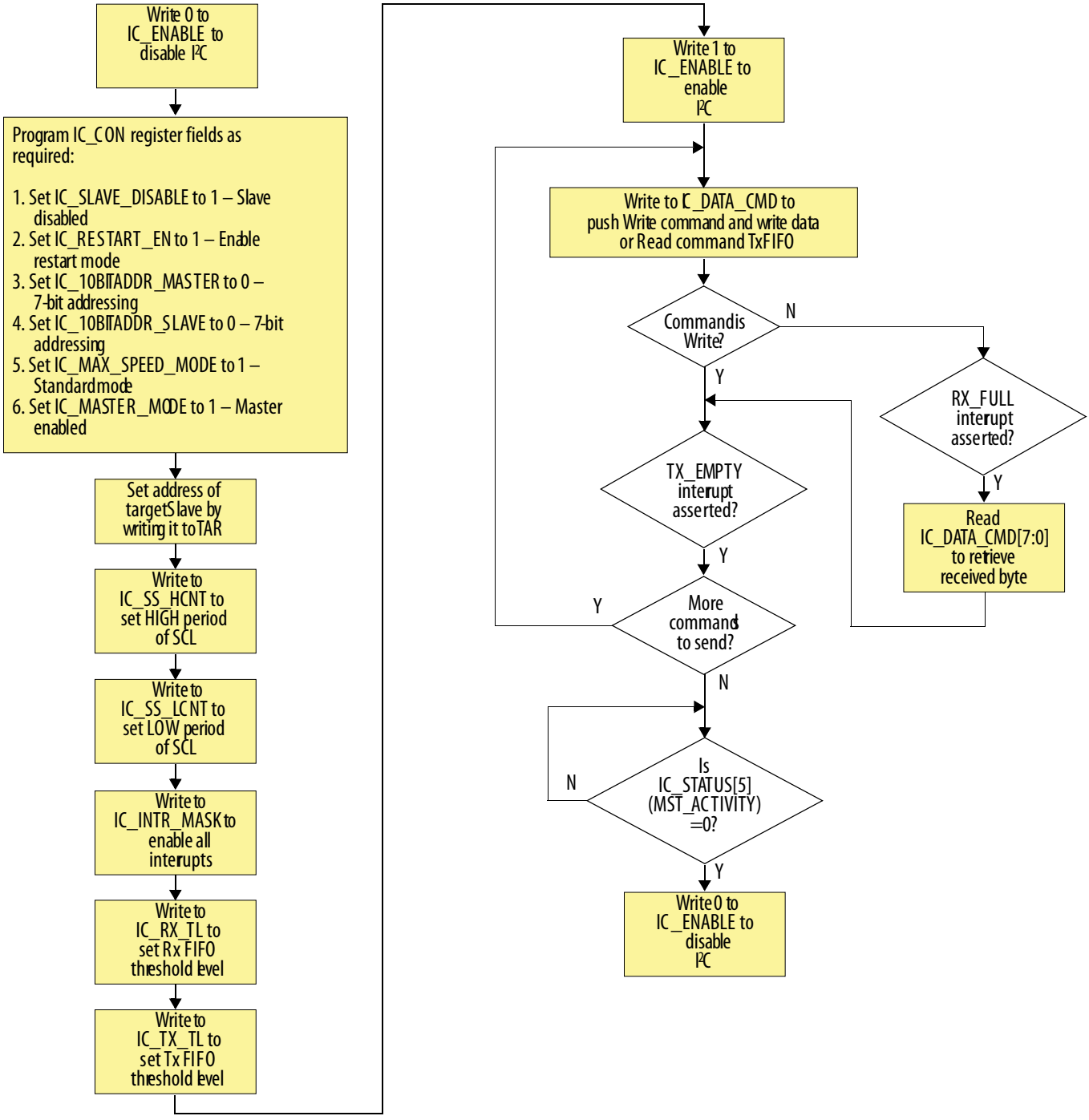
The I²C controller supports dynamic updating of the `IC_TAR` (bits 9:0) and `IC_10BITADDR_MASTER` (bit 12) bit fields of the `IC_TAR` register. You can dynamically write to the `IC_TAR` register provided the following conditions are met: †

- The I²C controller is not enabled (`IC_ENABLE`=0); †
- The I²C controller is enabled (`IC_ENABLE`=1); AND I²C controller is NOT engaged in any Master (TX, RX) operation (`IC_STATUS[5]`=0); AND I²C controller is enabled to operate in Master mode (`IC_CON[0]`=1); AND there are no entries in the TX FIFO (`IC_STATUS[2]`=1) †

Master Transmit and Master Receive

The I²C controller supports switching back and forth between reading and writing dynamically. To transmit data, write the data to be written to the lower byte of the I²C Rx/Tx Data Buffer and Command Register (`IC_DATA_CMD`). The CMD bit [8] should be written to 0 for I²C write operations. Subsequently, a read command may be issued by writing "don't cares" to the lower byte of the `IC_DATA_CMD` register, and a 1 should be written to the CMD bit.†

Master Programming Model



Disabling the I2C Controller

The register IC_ENABLE_STATUS is added to allow software to unambiguously determine when the hardware has completely shutdown in response to the IC_ENABLE register being set from 1 to 0. †

1. Define a timer interval (ti2c_poll) equal to the 10 times the signaling period for the highest I2C transfer speed used in the system and supported by the I2C controller. For example, if the highest I2C transfer mode is 400 Kbps, then ti2c_poll is 25 us. †
2. Define a maximum time-out parameter, MAX_T_POLL_COUNT, such that if any repeated polling operation exceeds this maximum value, an error is reported. †
3. Execute a blocking thread/process/function that prevents any further I2C master transactions to be started by software, but allows any pending transfers to be completed.
   - This step can be ignored if the I2C controller is programmed to operate as an I2C slave only. †
4. The variable POLL_COUNT is initialized to zero. †
5. Set IC_ENABLE to 0. †
6. Read the IC_ENABLE_STATUS register and test the IC_EN bit (bit 0). Increment POLL_COUNT by one. If POLL_COUNT >= MAX_T_POLL_COUNT, exit with the relevant error code. †
7. If IC_ENABLE_STATUS[0] is 1, then sleep for ti2c_poll and proceed to the previous step. Otherwise, exit with a relevant success code. †

Abort Transfer

The ABORT control bit of the IC_ENABLE register allows the software to relinquish the I2C bus before completing the issued transfer commands from the Tx FIFO. In response to an ABORT request, the controller issues the STOP condition over the I2C bus, followed by Tx FIFO flush. Aborting the transfer is allowed only in master mode of operation.†

1. Stop filling the Tx FIFO (IC_DATA_CMD) with new commands.†
2. When operating in DMA mode, disable the transmit DMA by setting TDMAE to 0. †
3. Set bit 1 of the IC_ENABLE register (ABORT) to 1.†
4. Wait for the M_TX_ABRT interrupt.†
5. Read the IC_TX_ABRT_SOURCE register to identify the source as ABRT_USER_ABRT.†

DMA Controller Operation

To enable the DMA controller interface on the I2C controller, you must write the DMA Control Register (IC_DMA_CR). Writing a 1 to the TDMAE bit field of IC_DMA_CR register enables the I2C controller transmit handshaking interface. Writing a 1 to the RDMAE bit field of the IC_DMA_CR register enables the I2C controller receive handshaking interface.†

The FIFO buffer depth (FIFO_DEPTH) for both the RX and TX buffers in the I2C controller is 64 entries.

Related Information

DMA Controller on page 16-1
For details about the DMA burst length microcode setup, refer to the *DMA controller* chapter.

Transmit FIFO Underflow

During I2C serial transfers, transmit FIFO requests are made to the DMA controller whenever the number of entries in the transmit FIFO is less than or equal to the value in DMA Transmit Data Level Register

(IC_DMA_TDLR), also known as the watermark level. The DMA controller responds by writing a burst of data to the transmit FIFO buffer, of length specified as DMA burst length. †

Note: Data should be fetched from the DMA often enough for the transmit FIFO to perform serial transfers continuously, that is, when the FIFO begins to empty, another DMA request should be triggered. Otherwise, the FIFO will run out of the data (underflow) causing the maser to stall the transfer by holding the SCL line low. To prevent this condition, you must set the watermark level correctly.†

Related Information

DMA Controller on page 16-1
For details about the DMA burst length microcode setup, refer to the *DMA controller* chapter.

Transmit Watermark Level

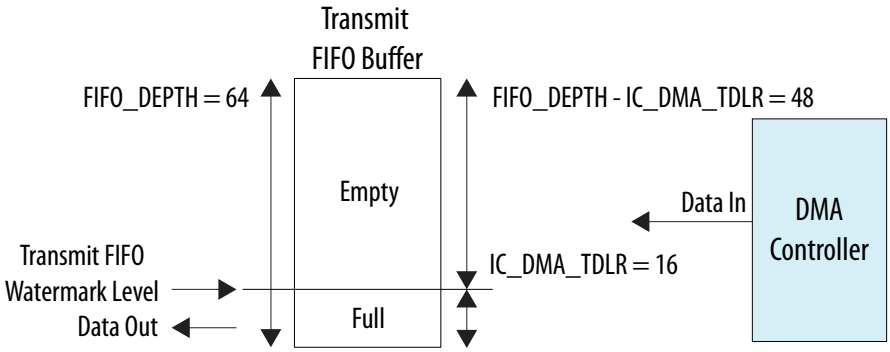
Consider the example where the assumption is made: †

DMA burst length = FIFO_DEPTH - IC_DMA_TDLR †

Here the number of data items to be transferred in a DMA burst is equal to the empty space in the transmit FIFO. Consider the following two different watermark level settings: †

- Case 1: IC_DMA_TDLR = 16: †
  - Transmit FIFO watermark level = IC_DMA_TDLR = 16: †
  - DMA burst length = FIFO_DEPTH - IC_DMA_TDLR = 48: †
  - I^2C transmit FIFO_DEPTH = 64: †
  - Block transaction size = 240: †

Figure 20-13: Transmit FIFO Watermark Level = 16



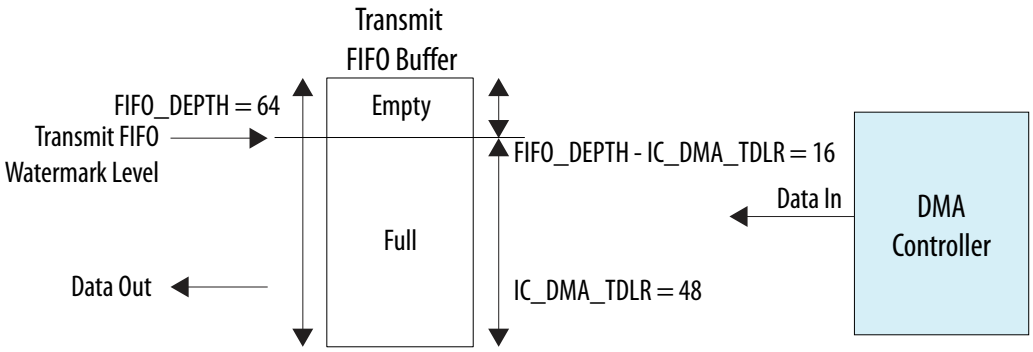
The number of burst transactions needed equals the block size divided by the number of data items per burst:

Block transaction size/DMA burst length = 240/48 = 5

The number of burst transactions in the DMA block transfer is 5. But the watermark level, IC_DMA_TDLR, is quite low. Therefore, the probability of transmit underflow is high where the I^2C serial transmit line needs to transmit data, but there is no data left in the transmit FIFO. This occurs because the DMA has not had time to service the DMA request before the FIFO becomes empty.

- Case 2: IC_DMA_TDLR = 48 †
  - Transmit FIFO watermark level = IC_DMA_TDLR = 48 †
  - DMA burst length = FIFO_DEPTH - IC_DMA_TDLR = 16 †
  - I²C transmit FIFO_DEPTH = 64 †
  - Block transaction size = 240 †

Figure 20-14: Transmit FIFO Watermark Level = 48



Number of burst transactions in block: †

Block transaction size/DMA burst length = 240/16 = 15 †

In this block transfer, there are 15 destination burst transactions in a DMA block transfer. But the watermark level, IC_ DMA_TDLR, is high. Therefore, the probability of I²C transmit underflow is low because the DMA controller has plenty of time to service the destination burst transaction request before the I²C transmit FIFO becomes empty. †

Thus, the second case has a lower probability of underflow at the expense of more burst transactions per block. This provides a potentially greater amount of bursts per block and worse bus utilization than the former case. †

Therefore, the goal in choosing a watermark level is to minimize the number of transactions per block, while at the same time keeping the probability of an underflow condition to an acceptable level. In practice, this is a function of the ratio of the rate at which the I²C transmits data to the rate at which the DMA can respond to destination burst requests. †

Transmit FIFO Overflow

Setting the DMA burst length to a value greater than the watermark level that triggers the DMA request might cause overflow when there is not enough space in the transmit FIFO to service the destination burst request. Therefore, the following equation must be adhered to in order to avoid overflow: †

DMA burst length <= FIFO_DEPTH - IC_DMA_TDLR

In case 2: IC_DMA_TDLR = 48, the amount of space in the transmit FIFO at the time of the burst request is made is equal to the DMA burst length. Thus, the transmit FIFO may be full, but not overflowed, at the completion of the burst transaction. †

Therefore, for optimal operation, DMA burst length should be set at the FIFO level that triggers a transmit DMA request; that is: †

DMA burst length = FIFO_DEPTH - IC_DMA_TDLR

Adhering to this equation reduces the number of DMA bursts needed for block transfer, and this in turn improves bus utilization. †

The transmit FIFO will not be full at the end of a DMA burst transfer if the I^2C controller has successfully transmitted one data item or more on the I^2C serial transmit line during the transfer. †

Receive FIFO Overflow

During I^2C serial transfers, receive FIFO requests are made to the DMA whenever the number of entries in the receive FIFO is at or above the DMA Receive Data Level Register, that is `IC_DMA_RDLR` + 1. This is known as the watermark level. The DMA responds by fetching a burst of data from the receive FIFO. †

Data should be fetched by the DMA often enough for the receive FIFO to accept serial transfers continuously, that is, when the FIFO begins to fill, another DMA transfer is requested. Otherwise the FIFO will fill with data (overflow). To prevent this condition, the user must set the watermark level correctly. †

Receive Watermark Level

Similar to choosing the transmit watermark level described earlier, the receive watermark level, `IC_DMA_RDLR` + 1, should be set to minimize the probability of overflow, as shown in the Receive FIFO Buffer diagram. It is a trade off between the number of DMA burst transactions required per block versus the probability of an overflow occurring. †

Receive FIFO Underflow

Setting the source transaction burst length greater than the watermark level can cause underflow where there is not enough data to service the source burst request. Therefore, the following equation must be adhered to avoid underflow: †

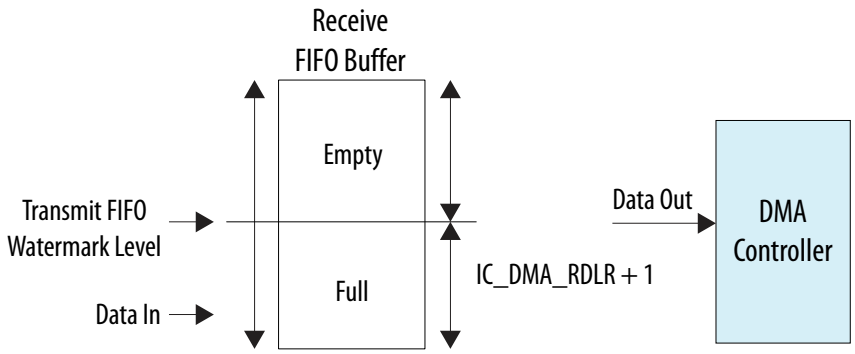
DMA burst length = `IC_DMA_RDLR` + 1

If the number of data items in the receive FIFO is equal to the source burst length at the time of the burst request is made, the receive FIFO may be emptied, but not underflowed, at the completion of the burst transaction. For optimal operation, DMA burst length should be set at the watermark level, `IC_DMA_RDLR` + 1. †

Adhering to this equation reduces the number of DMA bursts in a block transfer, which in turn can avoid underflow and improve bus utilization. †

Note: The receive FIFO will not be empty at the end of the source burst transaction if the I^2C controller has successfully received one data item or more on the I^2C serial receive line during the burst. †

Figure 20-15: Receive FIFO Buffer



I 2 C Controller Address Map and Register Definitions

For complete HPS address map and register definitions, refer to the **Arria 10 HPS Address Map and Register Definitions**.

i2c Address Map

Module Instance	Base Address	End Address
i_i2c_0_i2c	0xFFC02200	0xFFC022FF
i_i2c_1_i2c	0xFFC02300	0xFFC023FF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
ic_con on page 20-45	0x0	32	RW	0x7D	I2C Control Register
ic_tar on page 20-50	0x4	32	RW	0x1055	I2C Target Address Register
ic_sar on page 20-52	0x8	32	RW	0x55	I2C Slave Address Register
ic_data_cmd on page 20-53	0x10	32	RW	0x0	I2C Rx/Tx Data Buffer and Command Register
ic_ss_scl_hcnt on page 20-57	0x14	32	RW	0x190	Standard Speed I2C Clock SCL High Count Register
ic_ss_scl_lcnt on page 20-58	0x18	32	RW	0x1D6	Standard Speed I2C Clock SCL Low Count Register
ic_fs_scl_hcnt on page 20-59	0x1C	32	RW	0x3C	Fast Mode or Fast Mode Plus I2C Clock SCL High Count Register
ic_fs_scl_lcnt on page 20-60	0x20	32	RW	0x82	Fast Mode or Fast Mode Plus I2C Clock SCL Low Count Register
ic_intr_stat on page 20-62	0x2C	32	RO	0x0	I2C Interrupt Status Register
ic_intr_mask on page 20-69	0x30	32	RW	0x8FF	I2C Interrupt Mask Register

Register	Offset	Width	Access	Reset Value	Description
ic_raw_intr_stat on page 20-70	0x34	32	RO	0x0	I2C Raw Interrupt Status Register
ic_rx_tl on page 20-78	0x38	32	RW	0x0	I2C Receive FIFO Threshold Register
ic_tx_tl on page 20-79	0x3C	32	RW	0x0	I2C Transmit FIFO Threshold Register
ic_clr_intr on page 20-80	0x40	32	RO	0x0	Clear Combined and Individual Interrupt Register
ic_clr_rx_under on page 20-80	0x44	32	RO	0x0	Clear RX_UNDER Interrupt Register
ic_clr_rx_over on page 20-81	0x48	32	RO	0x0	Clear RX_OVER Interrupt Register
ic_clr_tx_over on page 20-82	0x4C	32	RO	0x0	Clear TX_OVER Interrupt Register
ic_clr_rd_req on page 20-83	0x50	32	RO	0x0	Clear RD_REQ Interrupt Register
ic_clr_tx_abrt on page 20-84	0x54	32	RO	0x0	Clear TX_ABRT Interrupt Register
ic_clr_rx_done on page 20-85	0x58	32	RO	0x0	Clear RX_DONE Interrupt Register
ic_clr_activity on page 20-86	0x5C	32	RO	0x0	Clear ACTIVITY Interrupt Register
ic_clr_stop_det on page 20-87	0x60	32	RO	0x0	Clear STOP_DET Interrupt Register
ic_clr_start_det on page 20-88	0x64	32	RO	0x0	Clear START_DET Interrupt Register
ic_clr_gen_call on page 20-89	0x68	32	RO	0x0	Clear GEN_CALL Interrupt Register
ic_enable on page 20-90	0x6C	32	RW	0x0	I2C Enable Register
ic_status on page 20-92	0x70	32	RO	0x6	I2C Status Register

Register	Offset	Width	Access	Reset Value	Description
ic_txflr on page 20-96	0x74	32	RO	0x0	I2C Transmit FIFO Level Register
ic_rxflr on page 20-97	0x78	32	RO	0x0	I2C Receive FIFO Level Register
ic_sda_hold on page 20-98	0x7C	32	RW	0x1	I2C SDA Hold Time Length Register
ic_tx_abrt_source on page 20-99	0x80	32	RO	0x0	I2C Transmit Abort Source Register
ic_slv_data_nack_only on page 20-104	0x84	32	RW	0x0	Generate Slave Data NACK Register.
ic_dma_cr on page 20-105	0x88	32	RW	0x0	DMA Control Register
ic_dma_tdlr on page 20-107	0x8C	32	RW	0x0	DMA Transmit Data Level Register
ic_dma_rdlr on page 20-108	0x90	32	RW	0x0	I2C Receive Data Level Register
ic_sda_setup on page 20-109	0x94	32	RW	0x64	I2C SDA Setup Register
ic_ack_general_call on page 20-110	0x98	32	RW	0x1	I2C ACK General Call Register
ic_enable_status on page 20-111	0x9C	32	RO	0x0	I2C Enable Status Register
ic_comp_param_1 on page 20-115	0xF4	32	RO	0x3F3FEA	Component Parameter Register 1
ic_comp_version on page 20-117	0xF8	32	RO	0x3132312A	I2C Component Version Register
ic_comp_type on page 20-118	0xFC	32	RO	0x44570140	I2C Component Type Register
ic_fs_spklen on page 20-119	0xA0	32	RW	0x2	I2C SS, FS or FM+ spike suppression limit
ic_clr_restart_det on page 20-120	0xA8	32	RO	0x0	Clear RESTART_DET Interrupt Register

i2c Summary

Module Instance	Base Address
i_i2c_0_i2c	0xFFC02200
i_i2c_1_i2c	0xFFC02300

Register Address Offset	Bit Fields															
i_i2c_0_i2c																
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	rsvd_ic_con_31to10 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ic_con 0x0	rsvd_ic_con_31to9 RO 0x0							tx_empty_ctrl RW 0x0	stop_det_ifaddressed RW 0x0	ic_slave_disable RW 0x1	ic_restart_en RW 0x1	ic_10bitaddr_master RO 0x1	ic_10bitaddr_slave RW 0x1	speed RW 0x2		master_mode RW 0x1
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ic_tar 0x4	Reserved			ic_10bitaddr_master RW 0x1	special RW 0x0	gc_or_start RW 0x0	ic_tar RW 0x55									
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ic_sar 0x8	Reserved						ic_sar RW 0x55									
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ic_data_cmd 0x10	Reserved					restart WO 0x0	stop WO 0x0	cmd WO 0x0	dat RW 0x0							

Register Address Offset	Bit Fields															
ic_ss_scl_hcnt 0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ic_ss_scl_hcnt RW 0x190															
ic_ss_scl_lcnt 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ic_ss_scl_lcnt RW 0x1D6															
ic_fs_scl_hcnt 0x1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ic_fs_scl_hcnt RW 0x3C															
ic_fs_scl_lcnt 0x20	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ic_fs_scl_lcnt RW 0x82															
ic_intr_stat 0x2C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		r_master_on_hold RO 0x0	r_restart_det RO 0x0	r_gen_call RO 0x0	r_start_det RO 0x0	r_stop_det RO 0x0	r_activity RO 0x0	r_rx_done RO 0x0	r_tx_abrt RO 0x0	r_rd_req RO 0x0	r_tx_empty RO 0x0	r_tx_over RO 0x0	r_rx_full RO 0x0	r_rx_over RO 0x0	r_rx_under RO 0x0
ic_intr_mask 0x30	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		m_master_on_hold RW 0x0	m_restart_det RW 0x0	m_gen_call RW 0x1	m_start_det RW 0x0	m_stop_det RW 0x0	m_activity RW 0x0	m_rx_done RW 0x1	m_tx_abrt RW 0x1	m_rd_req RW 0x1	m_tx_empty RW 0x1	m_tx_over RW 0x1	m_rx_full RW 0x1	m_rx_over RW 0x1	m_rx_under RW 0x1

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ic_raw_intr_stat 0x34	Reserved		master_on_hold RO 0x0	restart_det RO 0x0	gen_call RO 0x0	start_det RO 0x0	stop_det RO 0x0	activity RO 0x0	rx_done RO 0x0	tx_abrt RO 0x0	rd_req RO 0x0	tx_empty RO 0x0	tx_over RO 0x0	rx_full RO 0x0	rx_over RO 0x0	rx_under RO 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
ic_rx_tl 0x38	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								rx_tl RW 0x0							
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
ic_tx_tl 0x3C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								tx_tl RW 0x0							
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
ic_clr_intr 0x40	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															clr_intr RO 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
ic_clr_rx_under 0x44	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															clr_rx_under RO 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
ic_clr_rx_over 0x48	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															clr_rx_over RO 0x0

Register Address Offset	Bit Fields															
ic_clr_tx_over 0x4C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															clr_tx_over RO 0x0
ic_clr_rd_req 0x50	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															clr_rd_req RO 0x0
ic_clr_tx_abrt 0x54	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															clr_tx_abort RO 0x0
ic_clr_rx_done 0x58	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															clr_rx_done RO 0x0
ic_clr_activity 0x5C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															clr_activity RO 0x0
ic_clr_stop_det 0x60	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															clr_stop_det RO 0x0

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
ic_clr_start_det 0x64	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															clr_start_det RO 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
ic_clr_gen_call 0x68	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															clr_gen_call RO 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
ic_enable 0x6C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														txabort RW 0x0	enable RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
ic_status 0x70	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved									slv_activity RO 0x0	mst_activity RO 0x0	rff RO 0x0	rfne RO 0x0	tfe RO 0x1	tfnf RO 0x1	activity RO 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
ic_txflr 0x74	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved									txflr RO 0x0						
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
ic_rxflr 0x78	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved									rxflr RO 0x0						

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ic_sda_hold 0x7C	Reserved								ic_sda_rx_hold RW 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ic_sda_tx_hold RW 0x1															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	tx_flush_cnt RO 0x0									rsvd_ic_tx_abrt_source_22to17 RO 0x0						abrt_user_abrt RO 0x0
ic_tx_abrt_source 0x80	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	abrt_slvrd_intx RO 0x0	abrt_slv_arblost RO 0x0	abrt_slvflush_txfifo RO 0x0	arb_lost RO 0x0	abrt_master_dis RO 0x0	abrt_10b_rd_norstrt RO 0x0	abrt_sbyte_norstrt RO 0x0	abrt_hs_norstrt RO 0x0	abrt_sbyte_ackdet RO 0x0	abrt_hs_ackdet RO 0x0	abrt_gcall_read RO 0x0	abrt_gcall_noack RO 0x0	abrt_txdata_noack RO 0x0	abrt_10addr2_noack RO 0x0	abrt_10addr1_noack RO 0x0	abrt_7b_addr_noack RO 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
ic_slv_data_nack_only 0x84	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															nack RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	rsvd_ic_dma_cr_31to2 RO 0x0															
ic_dma_cr 0x88	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_ic_dma_cr_31to2 RO 0x0														tdmae RW 0x0	rdmae RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
ic_dma_tdlr 0x8C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved										dmatdl RW 0x0					

Register Address Offset	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ic_dma_rdlr 0x90	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved										dmardl RW 0x0					
ic_sda_setup 0x94	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								sda_setup RW 0x64							
ic_ack_general_call 0x98	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	rsvd_ic_ack_gen_31to1 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_ic_ack_gen_31to1 RO 0x0															ack_gen_call RW 0x1
ic_enable_status 0x9C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													slv_rx_data_lost RO 0x0	slv_disabled_while_busy RO 0x0	ic_en RO 0x0
ic_comp_param_1 0xF4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved								tx_buffer_depth RO 0x3F							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rx_buffer_depth RO 0x3F								add_encoded_params RO 0x1	has_dma RO 0x1	intr_io RO 0x1	hc_count_values RO 0x0	max_speed_mode RO 0x2		apb_data_width RO 0x2	

Register Address Offset	Bit Fields															
ic_comp_version 0xF8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ic_comp_version RO 0x3132312A															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ic_comp_version RO 0x3132312A															
ic_comp_type 0xFC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ic_comp_type RO 0x44570140															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ic_comp_type RO 0x44570140															
ic_fs_spklen 0xA0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								ic_fs_spklen RW 0x2							
ic_clr_restart_det 0xA8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															clr_restart_det RO 0x0
i_i2c_1_i2c																
ic_con 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	rsvd_ic_con_31to10 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_ic_con_31to10 RO 0x0						rx_fifo_full_hld_ctrl RW 0x0	tx_empty_ctrl RW 0x0	stop_det_ifaddressed RW 0x0	ic_slave_disable RW 0x1	ic_restart_en RW 0x1	ic_10bitaddr_master RO 0x1	ic_10bitaddr_slave RW 0x1	speed RW 0x2		master_mode RW 0x1

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ic_tar 0x4	Reserved			ic_10bitaddr_master RW 0x1	special RW 0x0	gc_or_start RW 0x0	ic_tar RW 0x55									
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
ic_sar 0x8	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						ic_sar RW 0x55									
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
ic_data_cmd 0x10	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved					restart WO 0x0	stop WO 0x0	cmd WO 0x0	dat RW 0x0							
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
ic_ss_scl_hcnt 0x14	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ic_ss_scl_hcnt RW 0x190															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
ic_ss_scl_lcnt 0x18	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ic_ss_scl_lcnt RW 0x1D6															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
ic_fs_scl_hcnt 0x1C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ic_fs_scl_hcnt RW 0x3C															

Register Address Offset	Bit Fields															
ic_fs_scl_lcnt 0x20	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ic_fs_scl_lcnt RW 0x82															
ic_intr_stat 0x2C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		r_master_on_hold RO 0x0	r_restart_det RO 0x0	r_gen_call RO 0x0	r_start_det RO 0x0	r_stop_det RO 0x0	r_activity RO 0x0	r_rx_done RO 0x0	r_tx_abrt RO 0x0	r_rd_req RO 0x0	r_tx_empty RO 0x0	r_tx_over RO 0x0	r_rx_full RO 0x0	r_rx_over RO 0x0	r_rx_under RO 0x0
ic_intr_mask 0x30	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		m_master_on_hold RW 0x0	m_restart_det RW 0x0	m_gen_call RW 0x1	m_start_det RW 0x0	m_stop_det RW 0x0	m_activity RW 0x0	m_rx_done RW 0x1	m_tx_abrt RW 0x1	m_rd_req RW 0x1	m_tx_empty RW 0x1	m_tx_over RW 0x1	m_rx_full RW 0x1	m_rx_over RW 0x1	m_rx_under RW 0x1
ic_raw_intr_stat 0x34	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		master_on_hold RO 0x0	restart_det RO 0x0	gen_call RO 0x0	start_det RO 0x0	stop_det RO 0x0	activity RO 0x0	rx_done RO 0x0	tx_abrt RO 0x0	rd_req RO 0x0	tx_empty RO 0x0	tx_over RO 0x0	rx_full RO 0x0	rx_over RO 0x0	rx_under RO 0x0
ic_rx_tl 0x38	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								rx_tl RW 0x0							

Register Address Offset	Bit Fields															
ic_tx_tl 0x3C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								tx_tl RW 0x0							
ic_clr_intr 0x40	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															clr_intr RO 0x0
ic_clr_rx_under 0x44	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															clr_rx_under RO 0x0
ic_clr_rx_over 0x48	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															clr_rx_over RO 0x0
ic_clr_tx_over 0x4C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															clr_tx_over RO 0x0
ic_clr_rd_req 0x50	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															clr_rd_req RO 0x0

Register Address Offset	Bit Fields															
ic_clr_tx_abrt 0x54	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															clr_tx_abort RO 0x0
ic_clr_rx_done 0x58	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															clr_rx_done RO 0x0
ic_clr_activity 0x5C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															clr_activity RO 0x0
ic_clr_stop_det 0x60	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															clr_stop_det RO 0x0
ic_clr_start_det 0x64	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															clr_start_det RO 0x0
ic_clr_gen_call 0x68	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															clr_gen_call RO 0x0

Register Address Offset	Bit Fields															
ic_enable 0x6C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														txabort RW 0x0	enable RW 0x0
ic_status 0x70	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved									slv_activity RO 0x0	mst_activity RO 0x0	rff RO 0x0	rfne RO 0x0	tfe RO 0x1	tfnf RO 0x1	activity RO 0x0
ic_txflr 0x74	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved									txflr RO 0x0						
ic_rxflr 0x78	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved									rxflr RO 0x0						
ic_sda_hold 0x7C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved								ic_sda_rx_hold RW 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ic_sda_tx_hold RW 0x1															

Register Address Offset	Bit Fields															
ic_tx_abrt_source 0x80	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	tx_flush_cnt RO 0x0									rsvd_ic_tx_abrt_source_22to17 RO 0x0						abrt_user_abrt RO 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	abrt_slvrd_intx RO 0x0	abrt_slv_arblost RO 0x0	abrt_slvflush_txfifo RO 0x0	arb_lost RO 0x0	abrt_master_dis RO 0x0	abrt_10b_rd_norstrt RO 0x0	abrt_sbyte_norstrt RO 0x0	abrt_hs_norstrt RO 0x0	abrt_sbyte_ackdet RO 0x0	abrt_hs_ackdet RO 0x0	abrt_gcall_read RO 0x0	abrt_gcall_noack RO 0x0	abrt_txdata_noack RO 0x0	abrt_10addr2_noack RO 0x0	abrt_10addr1_noack RO 0x0	abrt_7b_addr_noack RO 0x0
ic_slv_data_nack_only 0x84	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															nack RW 0x0
ic_dma_cr 0x88	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	rsvd_ic_dma_cr_31to2 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_ic_dma_cr_31to2 RO 0x0														tdmae RW 0x0	rdmae RW 0x0
ic_dma_tdlr 0x8C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved										dmatdl RW 0x0					
ic_dma_rdlr 0x90	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved										dmardl RW 0x0					

Register Address Offset	Bit Fields															
ic_sda_setup 0x94	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								sda_setup RW 0x64							
ic_ack_general_call 0x98	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	rsvd_ic_ack_gen_31to1 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_ic_ack_gen_31to1 RO 0x0															ack_gen_call RW 0x1
ic_enable_status 0x9C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													slv_rx_data_lost RO 0x0	slv_disabled_while_busy RO 0x0	ic_en RO 0x0
ic_comp_param_1 0xF4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved								tx_buffer_depth RO 0x3F							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rx_buffer_depth RO 0x3F								add_encoded_params RO 0x1	has_dma RO 0x1	intr_io RO 0x1	hc_count_values RO 0x0	max_speed_mode RO 0x2		apb_data_width RO 0x2	
ic_comp_version 0xF8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ic_comp_version RO 0x3132312A															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ic_comp_version RO 0x3132312A															

Register Address Offset	Bit Fields															
ic_comp_type 0xFC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ic_comp_type RO 0x44570140															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ic_comp_type RO 0x44570140															
ic_fs_spklen 0xA0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								ic_fs_spklen RW 0x2							
ic_clr_restart_det 0xA8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															clr_restart_det RO 0x0

ic_con

I2C Control Register. This register can be written only when the IC_ENABLE[0] register is set to 0. Writes at other times have no effect.

Module Instance	Base Address	Register Address
i_i2c_0_i2c	0xFFC02200	0xFFC02200
i_i2c_1_i2c	0xFFC02300	0xFFC02300

Offset: 0x0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
rsvd_ic_con_31to10 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rsvd_ic_con_31to9 RO 0x0							tx_empty_ctrl RW 0x0	stop_det_ifaddressed RW 0x0	ic_slave_disable RW 0x1	ic_restart_en RW 0x1	ic_10bitaddr_master RO 0x1	ic_10bitaddr_slave RW 0x1	speed RW 0x2		master_mode RW 0x1

ic_con Fields

Bit	Name	Description	Access	Reset
31:9	rsvd_ic_con_31to9	Reserved bits [31:9] - Read Only	RO	0x0
8	tx_empty_ctrl	This bit controls the generation of the TX_EMPTY interrupt, as described in the IC_RAW_INTR_STAT register. Reset value: 0x0.	RW	0x0
7	stop_det_ifaddressed	In slave mode: 1: issues the STOP_DET interrrupt only when it is addressed. 0: issues the STOP_DET irrespec-tive of whether it’s addressed or not. Dependencies: This register bit value is applicable in the slave mode only (MASTER_MODE = 1’b0) Reset value: 0x0 NOTE: During a general call address, this slave does not issue the STOP_DET interrupt if STOP_DET_IF_ADDRESSED = 1'b1, even if the slave responds to the general call address by generating ACK. The STOP_DET interrupt is generated only when the transmitted address matches the slave address (SAR).	RW	0x0

Bit	Name	Description	Access	Reset
6	ic_slave_disable	This bit controls whether I2C has its slave disabled. Once the presetn signal is applied, then this bit is set to 1. When this bit is set (slave is disabled), the I2C module functions only as a master and does not perform any action that requires a slave. 0: slave is enabled 1: slave is disabled NOTE: Software should ensure that if this bit is written with 0, then bit 0 should also be written with a 0. **Value** **Description** 0x0 ENABLE 0x1 DISABLE	RW	0x1

Bit	Name	Description	Access	Reset
5	ic_restart_en	Determines whether RESTART conditions may be sent when acting as a master. Some older slaves do not support handling RESTART conditions; however, RESTART conditions are used in several DW_apb_i2c operations. 0: disable 1: enable When RESTART is disabled, the master is prohibited from performing the following functions: - Change direction within a transfer (split) - Send a START BYTE - High-speed mode operation - Combined format transfers in 7-bit addressing modes - Read operation with a 10-bit address - Send multiple bytes per transfer By replacing RESTART condition followed by a STOP and a subsequent START condition, split operations are broken down into multiple I2C transfers. If the above operations are performed, it will result in setting bit 6 (TX_ABRT) of the IC_RAW_INTR_STAT register. Reset value: 0x1 **Value** **Description** 0x0 DISABLE 0x1 ENABLE	RW	0x1
4	ic_10bitaddr_master	The bit is read-only. 0: 7-bit addressing 1: 10-bit addressing Reset value: 0x1 **Value** **Description** 0x0 MSTADDR7BIT 0x1 MSTADDR10BIT	RO	0x1

Bit	Name	Description	Access	Reset
3	ic_10bitaddr_slave	When acting as a slave, this bit controls whether the I2C module responds to 7- or 10-bit addresses. 0: 7-bit addressing. The I2C module ignores transactions that involve 10-bit addressing; for 7-bit addressing, only the lower 7 bits of the IC_SAR register are compared. 1: 10-bit addressing. The I2C module responds to only 10-bit addressing transfers that match the full 10 bits of the IC_SAR register. Reset value: 0x1 **Value** **Description** 0x0 SLVADDR7BIT 0x1 SLVADDR10BIT	RW	0x1
2:1	speed	These bits control at which speed the I2C operates; its setting is relevant only if one is operating the I2C module in master mode. Hardware protects against illegal values being programmed by software. This register should be programmed only with a value in the range of 1 to 400 kbits/s; otherwise, hardware updates this register with the value of 400 kbits/s. 1: standard mode (100 kbit/s) 2: fast mode (<=400 kbit/s) or fast mode plus (<=1000Kbit/s) Reset value: 0x2 **Value** **Description** 0x1 STANDARD 0x2 FAST	RW	0x2

Bit	Name	Description	Access	Reset
0	master_mode	This bit controls whether the I2C master is enabled. 0: master disabled 1: master enabled Reset value: 0x1 NOTE: Software should ensure that if this bit is written with '1' then bit 6 should also be written with a '1'. **Value** **Description** 0x0 DISABLE 0x1 ENABLE	RW	0x1

ic_tar

```
Name: I2C Target Address Register

All bits can be dynamically updated as long as any set of the
following
conditions are true:
- The I2C is NOT enabled (IC_ENABLE[0] is set to 0);
   or
- The I2C is enabled (IC_ENABLE[0]=1);
   AND
  the I2C is NOT engaged in any Master (tx, rx) operation
(IC_STATUS[5]=0);
   AND
  I2C is enabled to operate in Master mode (IC_CON[0]=1);
   AND
  there are NO entries in the TX FIFO (IC_STATUS[2]=1)
```

Module Instance	Base Address	Register Address
i_i2c_0_i2c	0xFFC02200	0xFFC02204
i_i2c_1_i2c	0xFFC02300	0xFFC02304

Offset: 0x4

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			ic_10bit addr_master RW 0x1	special RW 0x0	gc_or_start RW 0x0	ic_tar RW 0x55									

ic_tar Fields

Bit	Name	Description	Access	Reset
12	ic_10bitaddr_master	This bit controls whether the I2C starts its transfers in 7- or 10-bit addressing mode when acting as a master. 0: 7-bit addressing 1: 10-bit addressing Reset value: 0x1 **Value** **Description** 0x0 START7 0x1 START10	RW	0x1
11	special	This bit indicates whether software performs a General Call or START BYTE command. 0: ignore bit 10 GC_OR_START and use IC_TAR normally 1: perform special I2C command as specified in GC_OR_START bit Reset value: 0x0 **Value** **Description** 0x0 GENCALL 0x1 STARTBYTE	RW	0x0

Bit	Name	Description	Access	Reset
10	gc_or_start	If bit 11 (SPECIAL) is set to 1, then this bit indicates whether a General Call or START byte command is to be performed by the I2C. 0: General Call Address after issuing a General Call, only writes may be performed. Attempting to issue a read command results in setting bit 6 (TX_ABRT) of the IC_RAW_INTR_STAT register. The I2C remains in General Call mode until the SPECIAL bit value (bit 11) is cleared. 1: START BYTE Reset value: 0x0 **Value / Description** 0x0 GENCALL 0x1 STARTBYTE	RW	0x0
9:0	ic_tar	This is the target address for any master transaction. When transmitting a General Call, these bits are ignored. To generate a START BYTE, the CPU needs to write only once into these bits. Reset value: 0x55 If the IC_TAR and IC_SAR are the same, loopback exists but the FIFOs are shared between master and slave, so full loopback is not feasible. Only one direction loopback mode is supported (simplex), not duplex. A master cannot transmit to itself; it can transmit to only a slave.	RW	0x55

ic_sar

Name: I2C Slave Address Register

Module Instance	Base Address	Register Address
i_i2c_0_i2c	0xFFC02200	0xFFC02208
i_i2c_1_i2c	0xFFC02300	0xFFC02308

Offset: 0x8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						ic_sar RW 0x55									

ic_sar Fields

Bit	Name	Description	Access	Reset
9:0	ic_sar	The IC_SAR holds the slave address when the I2C is operating as a slave. For 7-bit addressing, only IC_SAR[6:0] is used. This register can be written only when the I2C interface is disabled, which corresponds to the IC_ENABLE[0] register being set to 0. Writes at other times have no effect. Note The default values cannot be any of the reserved address locations: that is, 0x00 to 0x07, or 0x78 to 0x7f. The correct operation of the device is not guaranteed if you program the IC_SAR or IC_TAR to a reserved value. Reset value: 0x55	RW	0x55

ic_data_cmd

```
          I2C Rx/Tx Data Buffer and Command Register;
       this is the register the CPU writes to when
       filling the TX FIFO and the CPU reads from when
       retrieving bytes from RX FIFO
Size:
11 bits (writes), 8 bits (read)
```

Module Instance	Base Address	Register Address
i_i2c_0_i2c	0xFFC02200	0xFFC02210
i_i2c_1_i2c	0xFFC02300	0xFFC02310

Offset: 0x10

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					restart WO 0x0	stop WO 0x0	cmd WO 0x0	dat RW 0x0							

ic_data_cmd Fields

Bit	Name	Description	Access	Reset
10	restart	This bit controls whether a RESTART is issued before the byte is sent or received. 1 - If IC_RESTART_EN is 1, a RESTART is issued before the data is sent/received (according to the value of CMD), regardless of whether or not the transfer direction is changing from the previous command; if IC_RESTART_EN is 0, a STOP followed by a START is issued instead. 0 - If IC_RESTART_EN is 1, a RESTART is issued only if the transfer direction is changing from the previous command; if IC_RESTART_EN is 0, a STOP followed by a START is issued instead. Reset value: 0x0	WO	0x0
9	stop	This bit controls whether a STOP is issued after the byte is sent or received. 1 - STOP is issued after this byte, regardless of whether or not the Tx FIFO is empty. If the Tx FIFO is not empty, the master immediately tries to start a new transfer by issuing a START and arbitrating for the bus. 0 - STOP is not issued after this byte, regardless of whether or not the Tx FIFO is empty. If the Tx FIFO is not empty, the master continues the current transfer by sending/receiving data bytes according to the value of the CMD bit. If the Tx FIFO is empty, the master holds the SCL line low and stalls the bus until a new command is available in the Tx FIFO. Reset value: 0x0	WO	0x0

Bit	Name	Description	Access	Reset
8	cmd	This bit controls whether a read or a write is performed. This bit does not control the direction when the I2C acts as a slave. It controls only the direction when it acts as a master. 1 = Read 0 = Write When a command is entered in the TX FIFO, this bit distinguishes the write and read commands. In slave-receiver mode, this bit is a 'don't care' because writes to this register are not required. In slave-transmitter mode, a '0' indicates that CPU data is to be transmitted and as DAT or IC_DATA_CMD[7:0]. When programming this bit, you should remember the following: attempting to perform a read operation after a General Call command has been sent results in a TX_ABRT interrupt (bit 6 of the IC_RAW_INTR_STAT register), unless bit 11 (SPECIAL) in the IC_TAR register has been cleared. If a '1' is written to this bit after receiving a RD_REQ interrupt, then a TX_ABRT interrupt occurs. NOTE: It is possible that while attempting a master I2C read transfer, a RD_REQ interrupt may have occurred simultaneously due to a remote I2C master addressing the I2C module. In this type of scenario, the I2C mmodule ignores the IC_DATA_CMD write, generates a TX_ABRT interrupt, and waits to service the RD_REQ interrupt. Reset value: 0x0 **Value** **Description** 0x0 WR	WO	0x0

Bit	Name	Description	Access	Reset
		Value Description 0x1 RD		
7:0	dat	This register contains the data to be transmitted or received on the I2C bus. If you are writing to this register and want to perform a read, bits 7:0 (DAT) are ignored by the I2C module. However, when you read this register, these bits return the value of data received on the I2C interface. Reset value: 0x0	RW	0x0

ic_ss_scl_hcnt

Name: Standard Speed I2C Clock SCL High Count Register

Module Instance	Base Address	Register Address
i_i2c_0_i2c	0xFFC02200	0xFFC02214
i_i2c_1_i2c	0xFFC02300	0xFFC02314

Offset: 0x14

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ic_ss_scl_hcnt RW 0x190															

ic_ss_scl_hcnt Fields

Bit	Name	Description	Access	Reset
15:0	ic_ss_scl_hcnt	This register must be set before any I2C bus transaction can take place to ensure proper I/O timing. This register sets the SCL clock high-period count for standard speed. This register can be written only when the I2C interface is disabled which corresponds to the IC_ENABLE[0] register being set to 0. Writes at other times have no effect. The minimum valid value is 6; hardware prevents values less than this being written, and if attempted results in 6 being set. NOTE: This register must not be programmed to a value higher than 65525, because the I2C module uses a 16-bit counter to flag an I2C bus idle condition when this counter reaches a value of 0x019A.	RW	0x190

ic_ss_scl_lcnt

Standard Speed I2C Clock SCL Low Count Register

Module Instance	Base Address	Register Address
i_i2c_0_i2c	0xFFC02200	0xFFC02218
i_i2c_1_i2c	0xFFC02300	0xFFC02318

Offset: 0x18

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ic_ss_scl_lcnt RW 0x1D6															

ic_ss_scl_lcnt Fields

Bit	Name	Description	Access	Reset
15:0	ic_ss_scl_lcnt	This register must be set before any I2C bus transaction can take place to ensure proper I/O timing. This register sets the SCL clock low period count for standard speed. This register can be written only when the I2C interface is disabled which corresponds to the IC_ENABLE[0] register being set to 0. Writes at other times have no effect. The minimum valid value is 8; hardware prevents values less than this being written, and if attempted, results in 8 being set.	RW	0x1D6

ic_fs_scl_hcnt

Name: Fast Mode or Fast Mode Plus I2C Clock SCL High Count Register

Module Instance	Base Address	Register Address
i_i2c_0_i2c	0xFFC02200	0xFFC0221C
i_i2c_1_i2c	0xFFC02300	0xFFC0231C

Offset: 0x1C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ic_fs_scl_hcnt RW 0x3C															

ic_fs_scl_hcnt Fields

Bit	Name	Description	Access	Reset
15:0	ic_fs_scl_hcnt	This register must be set before any I2C bus transaction can take place to ensure proper I/O timing. This register sets the SCL clock high-period count for fast mode or fast mode plus. It is used in high-speed mode to send the Master Code and START BYTE or General CALL. This register can be written only when the I2C interface is disabled, which corresponds to the IC_ENABLE[0] register being set to 0. Writes at other times have no effect. The minimum valid value is 6; hardware prevents values less than this being written, and if attempted results in 6 being set. Reset value: 0x003C	RW	0x3C

ic_fs_scl_lcnt

Fast Mode or Fast Mode Plus I2C Clock SCL Low Count Register

Module Instance	Base Address	Register Address
i_i2c_0_i2c	0xFFC02200	0xFFC02220

Module Instance	Base Address	Register Address
i_i2c_1_i2c	0xFFC02300	0xFFC02320

Offset: 0x20

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ic_fs_scl_lcnt RW 0x82															

ic_fs_scl_lcnt Fields

Bit	Name	Description	Access	Reset
15:0	ic_fs_scl_lcnt	This register must be set before any I2C bus transaction can take place to ensure proper I/O timing. This register sets the SCL clock low period count for fast speed. It is used in high-speed mode to send the Master Code and START BYTE or General CALL. This register can be written only when the I2C interface is disabled, which corresponds to the IC_ENABLE[0] register being set to 0. Writes at other times have no effect. The minimum valid value is 8; hardware prevents values less than this being written, and if attempted results in 8 being set. Reset value: 0x0082	RW	0x82

ic_intr_stat

```
Name: I2C Interrupt Status Register

Each bit in this register has a corresponding mask bit
in the IC_INTR_MASK register. These bits are cleared by reading the
matching
interrupt clear register. The unmasked raw versions of these bits are
available in the IC_RAW_INTR_STAT register.
```

Module Instance	Base Address	Register Address
i_i2c_0_i2c	0xFFC02200	0xFFC0222C
i_i2c_1_i2c	0xFFC02300	0xFFC0232C

Offset: 0x2C

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		r_master_on_hold RO 0x0	r_restart_det RO 0x0	r_gen_call RO 0x0	r_start_det RO 0x0	r_stop_det RO 0x0	r_activity RO 0x0	r_rx_done RO 0x0	r_tx_abrt RO 0x0	r_rd_req RO 0x0	r_tx_empty RO 0x0	r_tx_over RO 0x0	r_rx_full RO 0x0	r_rx_over RO 0x0	r_rx_under RO 0x0

ic_intr_stat Fields

Bit	Name	Description	Access	Reset
13	r_master_on_hold	Indicates whether master is holding the bus and TX FIFO is empty. Reset value: 0x0	RO	0x0

Bit	Name	Description	Access	Reset
12	r_restart_det	Indicates a RESTART condition has occurred on the I2C interface when the I2C module is operating in slave mode and addressed. Reset value: 0x0	RO	0x0
11	r_gen_call	Set only when a General Call address is received and it is acknowledged. It stays set until it is cleared either by disabling the I2C or when the CPU reads bit 0 of the IC_CLR_GEN_CALL register. The I2C stores the received data in the Rx buffer. Reset value: 0x0	RO	0x0
10	r_start_det	Indicates whether a START or RESTART condition has occurred on the I2C interface regardless of whether the I2C is operating in slave or master mode. Reset value: 0x0	RO	0x0

Bit	Name	Description	Access	Reset
9	r_stop_det	The behavior of the STOP_DET interrupt status differs based on the STOP_DET_IFADDRESSED selection in the IC_CON register When STOP_DET_IFADDRESSED =0 : Indicates whether a STOP condition has occurred on the I2C interface regardless of whether DW_apb_i2c is operating in slave or master mode. In slave mode, a STOP_DET interrupt is generated irrespective of whether the slave is addressed or not. When STOP_DET_IFADDRESSED = 1 : In Master Mode (MASTER_MODE = 1'b1) , indicates a STOP condition has occurred on the I2C interface. In Slave Mode (MASTER_MODE = 1'b0) ,STOP_DET interrupt is generated only if the slave is addressed. NOTE: During a general call address, this slave does not issue a STOP_DET interrupt if STOP_DET_ IFADDRESSED=1'b1, even if the slave responds to the general call address by generating ACK. The STOP_DET interrupt is generated only when the transmitted address matches the slave address (SAR). Reset value: 0x0	RO	0x0

Bit	Name	Description	Access	Reset
8	r_activity	This bit captures the I2C activity and stays set until it is cleared. There are four ways to clear it: - Disabling the I2C module - Reading the IC_CLR_ACTIVITY register - Reading the IC_CLR_INTR register - System reset Once this bit is set, it stays set unless one of the four methods is used to clear it. Even if the I2C module is idle, this bit remains set until cleared, indicating that there was activity on the bus. Reset value: 0x0	RO	0x0
7	r_rx_done	When the I2C module is acting as a slave-transmitter, this bit is set to 1 if the master does not acknowledge a transmitted byte. This occurs on the last byte of the transmission, indicating that the transmission is done. Reset value: 0x0	RO	0x0
6	r_tx_abrt	This bit indicates if the I2C module, as an I2C transmitter, is unable to complete the intended actions on the contents of the transmit FIFO. This situation can occur both as an I2C master or an I2C slave, and is referred to as a 'transmit abort'. When this bit is set to 1, the IC_TX_ABRT_SOURCE register indicates the reason why the transmit abort takes places. NOTE: The I2C flushes/resets/empties the TX FIFO whenever this bit is set. The TX FIFO remains in this flushed state until the register IC_CLR_TX_ABRT is read. Once this read is performed, the TX FIFO is then ready to accept more data bytes from the APB interface. Reset value: 0x0	RO	0x0

Bit	Name	Description	Access	Reset
5	r_rd_req	This bit is set to 1 when the I2C module is acting as a slave and another I2C master is attempting to read data from it. The I2C module holds the I2C bus in a wait state (SCL=0) until this interrupt is serviced, which means that the slave has been addressed by a remote master that is asking for data to be transferred. The processor must respond to this interrupt and then write the requested data to the IC_DATA_CMD register. This bit is set to 0 just after the processor reads the IC_CLR_RD_REQ register. Reset value: 0x0	RO	0x0

Bit	Name	Description	Access	Reset
4	r_tx_empty	The behavior of the TX_EMPTY interrupt status differs based on the TX_EMPTY_CTRL selection in the IC_CON register. When TX_EMPTY_CTRL = 0: This bit is set to 1 when the transmit buffer is at or below the threshold value set in the IC_TX_TL register. When TX_EMPTY_CTRL = 1: This bit is set to 1 when the transmit buffer is at or below the threshold value set in the IC_TX_TL register and the transmission of the address/data from the internal shift register for the most recently popped command is completed. It is automatically cleared by hardware when the buffer level goes above the threshold. When IC_ENABLE[0] is set to 0, the TX FIFO is flushed and held in reset. There the TX FIFO looks like it has no data within it, so this bit is set to 1, provided there is activity in the master or slave state machines. When there is no longer any activity, then with ic_en=0, this bit is set to 0. Reset value: 0x0	RO	0x0
3	r_tx_over	Set during transmit if the transmit buffer is filled to IC_TX_BUFFER_DEPTH and the processor attempts to issue another I2C command by writing to the IC_DATA_CMD register. When the module is disabled, this bit keeps its level until the master or slave state machines go into idle, and when ic_en goes to 0, this interrupt is cleared. Reset value: 0x0	RO	0x0

Bit	Name	Description	Access	Reset
2	r_rx_full	Set when the receive buffer reaches or goes above the RX_TL threshold in the IC_RX_TL register. It is automatically cleared by hardware when buffer level goes below the threshold. If the module is disabled (IC_ENABLE[0]=0), the RX FIFO is flushed and held in reset; therefore the RX FIFO is not full. So this bit is cleared once the IC_ENABLE bit 0 is programmed with a 0, regardless of the activity that continues. Reset value: 0x0	RO	0x0
1	r_rx_over	Set if the receive buffer is completely filled to the Rx buffer depth of 64 and an additional byte is received from an external I2C device. The I2C acknowledges this, but any data bytes received after the FIFO is full are lost. If the module is disabled (IC_ENABLE[0]=0), this bit keeps its level until the master or slave state machines go into idle, and when ic_en goes to 0, this interrupt is cleared. Reset value: 0x0	RO	0x0
0	r_rx_under	Set if the processor attempts to read the receive buffer when it is empty by reading from the IC_DATA_CMD register. If the module is disabled (IC_ENABLE[0]=0), this bit keeps its level until the master or slave state machines go into idle, and when ic_en goes to 0, this interrupt is cleared. Reset value: 0x0	RO	0x0

ic_intr_mask

Name: I2C Interrupt Mask Register

These bits mask their corresponding interrupt status bits. They are active high; a value of 0 prevents a bit from generating an interrupt.

Module Instance	Base Address	Register Address
i_i2c_0_i2c	0xFFC02200	0xFFC02230
i_i2c_1_i2c	0xFFC02300	0xFFC02330

Offset: 0x30

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		m_master_on_hold RW 0x0	m_restart_det RW 0x0	m_gen_call RW 0x1	m_start_det RW 0x0	m_stop_det RW 0x0	m_activity RW 0x0	m_rx_done RW 0x1	m_tx_abrt RW 0x1	m_rd_req RW 0x1	m_tx_empty RW 0x1	m_tx_over RW 0x1	m_rx_full RW 0x1	m_rx_over RW 0x1	m_rx_under RW 0x1

ic_intr_mask Fields

Bit	Name	Description	Access	Reset
13	m_master_on_hold	This bit masks the R_MASTER_ON_HOLD interrupt in IC_INTR_STAT register. Reset value: 0x0	RW	0x0
12	m_restart_det	This bit masks the R_RESTART_DET interrupt in IC_INTR_STAT register. Reset value: 0x0	RW	0x0
11	m_gen_call	This bit masks the R_GEN_CALL interrupt in IC_INTR_STAT register. Reset value: 0x1	RW	0x1

Bit	Name	Description	Access	Reset
10	m_start_det	This bit masks the R_START_DET interrupt in IC_INTR_STAT register. Reset value: 0x0	RW	0x0
9	m_stop_det	This bit masks the R_STOP_DET interrupt in IC_INTR_STAT register. Reset value: 0x0	RW	0x0
8	m_activity	This bit masks the R_ACTIVITY interrupt in IC_INTR_STAT register. Reset value: 0x0	RW	0x0
7	m_rx_done	This bit masks the R_RX_DONE interrupt in IC_INTR_STAT register. Reset value: 0x1	RW	0x1
6	m_tx_abrt	This bit masks the R_TX_ABRT interrupt in IC_INTR_STAT register. Reset value: 0x1	RW	0x1
5	m_rd_req	This bit masks the R_RD_REQ interrupt in IC_INTR_STAT register. Reset value: 0x1	RW	0x1
4	m_tx_empty	This bit masks the R_TX_EMPTY interrupt in IC_INTR_STAT register. Reset value: 0x1	RW	0x1
3	m_tx_over	This bit masks the R_TX_OVER interrupt in IC_INTR_STAT register. Reset value: 0x1	RW	0x1
2	m_rx_full	This bit masks the R_RX_FULL interrupt in IC_INTR_STAT register. Reset value: 0x1	RW	0x1
1	m_rx_over	This bit masks the R_RX_OVER interrupt in IC_INTR_STAT register. Reset value: 0x1	RW	0x1
0	m_rx_under	This bit masks the R_RX_UNDER interrupt in IC_INTR_STAT register. Reset value: 0x1	RW	0x1

ic_raw_intr_stat

I2C Raw Interrupt Status Register

Unlike the IC_INTR_STAT register, these bits are not masked so they

always show the true status of the I2C module.

Module Instance	Base Address	Register Address
i_i2c_0_i2c	0xFFC02200	0xFFC02234
i_i2c_1_i2c	0xFFC02300	0xFFC02334

Offset: 0x34

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		master_on_hold RO 0x0	restart_det RO 0x0	gen_call RO 0x0	start_det RO 0x0	stop_det RO 0x0	activity RO 0x0	rx_done RO 0x0	tx_abrt RO 0x0	rd_req RO 0x0	tx_empty RO 0x0	tx_over RO 0x0	rx_full RO 0x0	rx_over RO 0x0	rx_under RO 0x0

ic_raw_intr_stat Fields

Bit	Name	Description	Access	Reset
13	master_on_hold	Indicates whether master is holding the bus and TX FIFO is empty. Reset value: 0x0	RO	0x0

Bit	Name	Description	Access	Reset
12	restart_det	Indicates whether a RESTART condition has occurred on the I2C interface when it is operating in Slave mode and the slave is being addressed. (Note:Following are exceptions where the Restart interrupt will not get generated. In the case of a Startbyte transfer, where the Restart comes before the Address field as per the I2C protocol defined format, the Slave is still not in the addressed mode and hence will not generate the RESTART_DET interrupt.) Reset value: 0x0	RO	0x0
11	gen_call	Set only when a General Call address is received and it is acknowledged. It stays set until it is cleared either by disabling the I2C or when the CPU reads bit 0 of the IC_CLR_GEN_CALL register. The I2C stores the received data in the Rx buffer. Reset value: 0x0	RO	0x0
10	start_det	Indicates whether a START or RESTART condition has occurred on the I2C interface regardless of whether the I2C is operating in slave or master mode. Reset value: 0x0	RO	0x0

Bit	Name	Description	Access	Reset

9	stop_det	The behavior of the STOP_DET interrupt status differs based on the STOP_DET_IFADDRESSED selection in the IC_CON register
		When STOP_DET_IFADDRESSED =0
		  Indicates whether a STOP condition has occurred on the I2C interface regardless
		  of whether the I2C module is operating in slave or master mode. In slave mode,
		  a STOP_DET interrupt is generated irrespective of whether the
		  slave is addressed or not.
		When STOP_DET_IFADDRESSED = 1
		  In Master Mode (MASTER_MODE = 1), indicates a STOP condition has occured on the I2C interface.
		  In Slave Mode (MASTER_MODE = 0), a STOP_DET interrupt is generated only if the slave is addressed.
		
		NOTE: During a general call address, this slave does not issue a STOP_DET
		interrupt if STOP_DET_IFADDRESSED=1, even if the slave responds to the
		general call address by generating ACK. The STOP_DET interrupt is generated
		only when the transmitted address matches the slave address (SAR).
		Reset value: 0x0.
			RO	0x0

Bit	Name	Description	Access	Reset
8	activity	This bit captures the I2C activity and stays set until it is cleared. There are four ways to clear it: - Disabling the I2C module - Reading the IC_CLR_ACTIVITY register - Reading the IC_CLR_INTR register - System reset Once this bit is set, it stays set unless one of the four methods is used to clear it. Even if the the I2C module is idle, this bit remains set until cleared, indicating that there was activity on the bus. Reset value: 0x0	RO	0x0
7	rx_done	When the I2C is acting as a slave-transmitter, this bit is set to 1 if the master does not acknowledge a transmitted byte. This occurs on the last byte of the transmission, indicating that the transmission is done. Reset value: 0x0	RO	0x0
6	tx_abrt	This bit indicates if the I2C, as an I2C transmitter, is unable to complete the intended actions on the contents of the transmit FIFO. This situation can occur both as an I2C master or an I2C slave, and is referred to as a 'transmit abort'. When this bit is set to 1, the IC_TX_ABRT_SOURCE register indicates the reason why the transmit abort takes places. NOTE: The I2C flushes/resets/empties the TX FIFO whenever this bit is set. The TX FIFO remains in this flushed state until the register IC_CLR_TX_ABRT is read. Once this read is performed, the TX FIFO is then ready to accept more data bytes from the APB interface. Reset value: 0x0	RO	0x0

Bit	Name	Description	Access	Reset
5	rd_req	This bit is set to 1 when the I2C is acting as a slave and another I2C master is attempting to read data from it. The HPS I2C holds the I2C bus in a wait state (SCL=0) until this interrupt is serviced, which means that the slave has been addressed by a remote master that is asking for data to be transferred. The processor must respond to this interrupt and then write the requested data to the IC_DATA_CMD register. This bit is set to 0 just after the processor reads the IC_CLR_RD_REQ register. Reset value: 0x0	RO	0x0

Bit	Name	Description	Access	Reset
4	tx_empty	The behavior of the TX_EMPTY interrupt status differs based on the TX_EMPTY_CTRL selection in the IC_CON register. When TX_EMPTY_CTRL = 0: This bit is set to 1 when the transmit buffer is at or below the threshold value set in the IC_TX_TL register. When TX_EMPTY_CTRL = 1: This bit is set to 1 when the transmit buffer is at or below the threshold value. set in the IC_TX_TL register and the transmission of the address/ data from the internal shift register for the most recently popped command is completed. It is automatically cleared by hardware when the buffer level goes above the threshold. When IC_ENABLE[0] is set to 0, the TX FIFO is flushed and held in reset. There the TX FIFO looks like it has no data within it, so this bit is set to 1, provided there is activity in the master or slave state machines. When there is no longer any activity, then with ic_ en=0, this bit is set to 0. Reset value: 0x0.	RO	0x0
3	tx_over	Set during transmit if the transmit buffer is filled to its maximum TxFIFO buffer depth (64) and the processor attempts to issue another I2C command by writing to the IC_DATA_CMD register. When the module is disabled, this bit keeps its level until the master or slave state machines go into idle, and when ic_ en goes to 0, this interrupt is cleared. Reset value: 0x0	RO	0x0

Bit	Name	Description	Access	Reset
2	rx_full	Set when the receive buffer reaches or goes above the RX_TL threshold in the IC_RX_TL register. It is automatically cleared by hardware when buffer level goes below the threshold. If the module is disabled (IC_ENABLE[0]=0), the RX FIFO is flushed and held in reset; therefore the RX FIFO is not full. So this bit is cleared once the IC_ENABLE bit 0 is programmed with a 0, regardless of the activity that continues. Reset value: 0x0	RO	0x0
1	rx_over	Set if the receive buffer is completely filled to its maximum RxFIFO depth (64) and an additional byte is received from an external I2C device. The HPS I2C acknowledges this, but any data bytes received after the FIFO is full are lost. If the module is disabled (IC_ENABLE[0]=0), this bit keeps its level until the master or slave state machines go into idle, and when ic_en goes to 0, this interrupt is cleared. Reset value: 0x0	RO	0x0
0	rx_under	Set if the processor attempts to read the receive buffer when it is empty by reading from the IC_DATA_CMD register. If the module is disabled (IC_ENABLE[0]=0), this bit keeps its level until the master or slave state machines go into idle, and when ic_en goes to 0, this interrupt is cleared. Reset value: 0x0	RO	0x0

ic_rx_tl

Name: I2C Receive FIFO Threshold Register

Module Instance	Base Address	Register Address
i_i2c_0_i2c	0xFFC02200	0xFFC02238
i_i2c_1_i2c	0xFFC02300	0xFFC02338

Offset: 0x38

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								rx_tl RW 0x0							

ic_rx_tl Fields

Bit	Name	Description	Access	Reset
7:0	rx_tl	Receive FIFO Threshold Level Controls the level of entries (or above) that triggers the RX_FULL interrupt (bit 2 in IC_RAW_INTR_STAT register). The valid range is 0-255, with the additional restriction that hardware does not allow this value to be set to a value larger than the depth of the buffer. If an attempt is made to do that, the actual value set will be the maximum depth of the buffer. A value of 0 sets the threshold for 1 entry, and a value of 255 sets the threshold for 256 entries.	RW	0x0

ic_tx_tl

I2C Transmit FIFO Threshold Register

Module Instance	Base Address	Register Address
i_i2c_0_i2c	0xFFC02200	0xFFC0223C
i_i2c_1_i2c	0xFFC02300	0xFFC0233C

Offset: 0x3C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								tx_tl RW 0x0							

ic_tx_tl Fields

Bit	Name	Description	Access	Reset
7:0	tx_tl	Transmit FIFO Threshold Level Controls the level of entries (or below) that trigger the TX_EMPTY interrupt (bit 4 in IC_RAW_INTR_STAT register). The valid range is 0-255, with the additional restriction that it may not be set to value larger than the depth of the buffer. If an attempt is made to do that, the actual value set will be the maximum depth of the buffer. A value of 0 sets the threshold for 0 entries, and a value of 255 sets the threshold for 255 entries.	RW	0x0

ic_clr_intr

Clear Combined and Individual Interrupt Register

Module Instance	Base Address	Register Address
i_i2c_0_i2c	0xFFC02200	0xFFC02240
i_i2c_1_i2c	0xFFC02300	0xFFC02340

Offset: 0x40

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															clr_intr RO 0x0

ic_clr_intr Fields

Bit	Name	Description	Access	Reset
0	clr_intr	Read this register to clear the combined interrupt, all individual interrupts, and the IC_TX_ABRT_SOURCE register. This bit does not clear hardware clearable interrupts but software clearable interrupts. Refer to Bit 9 of the IC_TX_ABRT_SOURCE register for an exception to clearing IC_TX_ABRT_SOURCE. Reset value: 0x0	RO	0x0

ic_clr_rx_under

Clear RX_UNDER Interrupt Register

Module Instance	Base Address	Register Address
i_i2c_0_i2c	0xFFC02200	0xFFC02244
i_i2c_1_i2c	0xFFC02300	0xFFC02344

Offset: 0x44

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															clr_rx_under RO 0x0

ic_clr_rx_under Fields

Bit	Name	Description	Access	Reset
0	clr_rx_under	Read this register to clear the RX_UNDER interrupt (bit 0) of the IC_RAW_INTR_STAT register. Reset value: 0x0	RO	0x0

ic_clr_rx_over

Clear RX_OVER Interrupt Register

Module Instance	Base Address	Register Address
i_i2c_0_i2c	0xFFC02200	0xFFC02248
i_i2c_1_i2c	0xFFC02300	0xFFC02348

Offset: 0x48

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															clr_rx_over RO 0x0

ic_clr_rx_over Fields

Bit	Name	Description	Access	Reset
0	clr_rx_over	Read this register to clear the RX_OVER interrupt (bit 1) of the IC_RAW_INTR_STAT register. Reset value: 0x0	RO	0x0

ic_clr_tx_over

```
          Name: Clear TX_OVER Interrupt Register
Size: 1 bit
Address Offset: 0x4c
Read/Write Access: Read
```

Module Instance	Base Address	Register Address
i_i2c_0_i2c	0xFFC02200	0xFFC0224C
i_i2c_1_i2c	0xFFC02300	0xFFC0234C

Offset: 0x4C

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															clr_tx_over RO 0x0

ic_clr_tx_over Fields

Bit	Name	Description	Access	Reset
0	clr_tx_over	Read this register to clear the TX_OVER interrupt (bit 3) of the IC_RAW_INTR_STAT register. Reset value: 0x0	RO	0x0

ic_clr_rd_req

```
Name: Clear RD_REQ Interrupt Register
Size: 1 bit
Address Offset: 0x50
Read/Write Access: Read
```

Module Instance	Base Address	Register Address
i_i2c_0_i2c	0xFFC02200	0xFFC02250
i_i2c_1_i2c	0xFFC02300	0xFFC02350

Offset: 0x50

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															clr_rd_req RO 0x0

ic_clr_rd_req Fields

Bit	Name	Description	Access	Reset
0	clr_rd_req	Read this register to clear the RD_REQ interrupt (bit 5) of the IC_RAW_INTR_STAT register. Reset value: 0x0	RO	0x0

ic_clr_tx_abrt

```
          Name: Clear TX_ABRT Interrupt Register
Size: 1 bit
Address Offset: 0x54
Read/Write Access: Read
```

Module Instance	Base Address	Register Address
i_i2c_0_i2c	0xFFC02200	0xFFC02254
i_i2c_1_i2c	0xFFC02300	0xFFC02354

Offset: 0x54

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															clr_tx_abort RO 0x0

ic_clr_tx_abrt Fields

Bit	Name	Description	Access	Reset
0	clr_tx_abort	Read this register to clear the TX_ABRT interrupt (bit 6) of the IC_RAW_INTR_STAT register, and the IC_TX_ABRT_SOURCE register. This also releases the TX FIFO from the flushed/reset state, allowing more writes to the TX FIFO. Refer to Bit 9 of the IC_TX_ABRT_SOURCE register for an exception to clearing IC_TX_ABRT_SOURCE. Reset value: 0x0	RO	0x0

ic_clr_rx_done

```
Name: Clear RX_DONE Interrupt Register
Size: 1 bit
Address Offset: 0x58
Read/Write Access: Read
```

Module Instance	Base Address	Register Address
i_i2c_0_i2c	0xFFC02200	0xFFC02258
i_i2c_1_i2c	0xFFC02300	0xFFC02358

Offset: 0x58

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															clr_rx_done RO 0x0

ic_clr_rx_done Fields

Bit	Name	Description	Access	Reset
0	clr_rx_done	Read this register to clear the RX_DONE interrupt (bit 7) of the IC_RAW_INTR_STAT register. Reset value: 0x0	RO	0x0

ic_clr_activity

```
          Name: Clear ACTIVITY Interrupt Register
Size: 1 bit
Address Offset: 0x5c
Read/Write Access: Read
```

Module Instance	Base Address	Register Address
i_i2c_0_i2c	0xFFC02200	0xFFC0225C
i_i2c_1_i2c	0xFFC02300	0xFFC0235C

Offset: 0x5C

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															clr_activity RO 0x0

ic_clr_activity Fields

Bit	Name	Description	Access	Reset
0	clr_activity	Reading this register clears the ACTIVITY interrupt if the I2C is not active anymore. If the I2C module is still active on the bus, the ACTIVITY interrupt bit continues to be set. It is automatically cleared by hardware if the module is disabled and if there is no further activity on the bus. The value read from this register to get status of the ACTIVITY interrupt (bit 8) of the IC_RAW_INTR_STAT register. Reset value: 0x0	RO	0x0

ic_clr_stop_det

```
        Name: Clear STOP_DET Interrupt Register
Size: 1 bit
Address Offset: 0x60
Read/Write Access: Read
```

Module Instance	Base Address	Register Address
i_i2c_0_i2c	0xFFC02200	0xFFC02260
i_i2c_1_i2c	0xFFC02300	0xFFC02360

Offset: 0x60

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															clr_stop_det RO 0x0

ic_clr_stop_det Fields

Bit	Name	Description	Access	Reset
0	clr_stop_det	Read this register to clear the STOP_DET interrupt (bit 9) of the IC_RAW_INTR_STAT register. Reset value: 0x0	RO	0x0

ic_clr_start_det

```
Name: Clear START_DET Interrupt Register
Size: 1 bit
Address Offset: 0x64
Read/Write Access: Read
```

Module Instance	Base Address	Register Address
i_i2c_0_i2c	0xFFC02200	0xFFC02264
i_i2c_1_i2c	0xFFC02300	0xFFC02364

Offset: 0x64

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															clr_start_det RO 0x0

ic_clr_start_det Fields

Bit	Name	Description	Access	Reset
0	clr_start_det	Read this register to clear the START_DET interrupt (bit 10) of the IC_RAW_INTR_STAT register. Reset value: 0x0	RO	0x0

ic_clr_gen_call

```
          Name: Clear GEN_CALL Interrupt Register
Size: 1 bit
Address Offset: 0x68
Read/Write Access: Read
```

Module Instance	Base Address	Register Address
i_i2c_0_i2c	0xFFC02200	0xFFC02268
i_i2c_1_i2c	0xFFC02300	0xFFC02368

Offset: 0x68

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															clr_gen_call RO 0x0

ic_clr_gen_call Fields

Bit	Name	Description	Access	Reset
0	clr_gen_call	Read this register to clear the GEN_CALL interrupt (bit 11) of IC_RAW_INTR_STAT register. Reset value: 0x0	RO	0x0

ic_enable

I2C Enable Register

Module Instance	Base Address	Register Address
i_i2c_0_i2c	0xFFC02200	0xFFC0226C
i_i2c_1_i2c	0xFFC02300	0xFFC0236C

Offset: 0x6C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														txabort RW 0x0	enable RW 0x0

ic_enable Fields

Bit	Name	Description	Access	Reset
1	txabort	When set, the controller initiates the transfer abort. 0: ABORT not initiated or ABORT done 1: ABORT operation in progress The software can abort the I2C transfer in master mode by setting this bit. The software can set this bit only when ENABLE is already set; otherwise, the controller ignores any write to ABORT bit. The software cannot clear the ABORT bit once set. In response to an ABORT, the controller issues a STOP and flushes the Tx FIFO after completing the current transfer, then sets the TX_ABORT interrupt after the abort operation. The ABORT bit is cleared automatically after the abort operation. Reset value: 0x0	RW	0x0

Bit	Name	Description	Access	Reset
0	enable	Controls whether the I2C module is enabled. 0: Disables the I2C module (TX and RX FIFOs are held in an erased state) 1: Enables the I2C module Software can disable the I2C module while it is active. However, it is important that care be taken to ensure that the I2C module is disabled properly. When the I2C module is disabled, the following occurs: - The TX FIFO and RX FIFO get flushed. - Status bits in the IC_INTR_STAT register are still active until the I2C goes into IDLE state. If the module is transmitting, it stops as well as deletes the contents of the transmit buffer after the current transfer is complete. If the module is receiving, the I2C stops the current transfer at the end of the current byte and does not acknowledge the transfer. Reset value: 0x0 **Value** **Description** 0x0 DISABLE 0x1 ENABLE	RW	0x0

ic_status

```
          Name: I2C Status Register

This is a read-only register used to indicate the current
transfer status and FIFO status. The status register may be
read at any time. None of the bits in this register request
an interrupt.
When the I2C is disabled by writing 0 in bit 0 of the
IC_ENABLE register:
- Bits 1 and 2 are set to 1
- Bits 3 and 4 are set to 0
When the master or slave state machines goes to idle
```

```
and ic_en=0:
- Bits 5 and 6 are set to 0
```

Module Instance	Base Address	Register Address
i_i2c_0_i2c	0xFFC02200	0xFFC02270
i_i2c_1_i2c	0xFFC02300	0xFFC02370

Offset: 0x70

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									slv_activity RO 0x0	mst_activity RO 0x0	rff RO 0x0	rfne RO 0x0	tfe RO 0x1	tfnf RO 0x1	activity RO 0x0

ic_status Fields

Bit	Name	Description	Access	Reset
6	slv_activity	Slave FSM Activity Status. When the Slave Finite State Machine (FSM) is not in the IDLE state, this bit is set. 0: Slave FSM is in IDLE state so the Slave part of the I2C is not Active 1: Slave FSM is not in IDLE state so the Slave part of the I2C is Active Reset value: 0x0 **Value** **Description** 0x0 IDLE 0x1 NOTIDLE	RO	0x0

Bit	Name	Description	Access	Reset
5	mst_activity	Master FSM Activity Status. When the Master Finite State Machine (FSM) is not in the IDLE state, this bit is set. 0: Master FSM is in IDLE state so the Master part of the I2C is not Active 1: Master FSM is not in IDLE state so the Master part of the I2C is Active Note IC_STATUS[0]-that is, ACTIVITY bit-is the OR of SLV_ACTIVITY and MST_ACTIVITY bits. Reset value: 0x0 **Value** **Description** 0x0 IDLE 0x1 NOTIDLE	RO	0x0
4	rff	Receive FIFO Completely Full. When the receive FIFO is completely full, this bit is set. When the receive FIFO contains one or more empty location, this bit is cleared. 0: Receive FIFO is not full 1: Receive FIFO is full Reset value: 0x0 **Value** **Description** 0x0 NOTFULL 0x1 FULL	RO	0x0

Bit	Name	Description	Access	Reset
3	rfne	Receive FIFO Not Empty. This bit is set when the receive FIFO contains one or more entries; it is cleared when the receive FIFO is empty. 0: Receive FIFO is empty 1: Receive FIFO is not empty Reset value: 0x0 **Value** / **Description** 0x0 — EMPTY 0x1 — NOTEMPTY	RO	0x0
2	tfe	Transmit FIFO Completely Empty. When the transmit FIFO is completely empty, this bit is set. When it contains one or more valid entries, this bit is cleared. This bit field does not request an interrupt. 0: Transmit FIFO is not empty 1: Transmit FIFO is empty Reset value: 0x1 **Value** / **Description** 0x0 — NOTEMPTY 0x1 — EMPTY	RO	0x1
1	tfnf	Transmit FIFO Not Full. Set when the transmit FIFO contains one or more empty locations, and is cleared when the FIFO is full. 0: Transmit FIFO is full 1: Transmit FIFO is not full Reset value: 0x1 **Value** / **Description** 0x0 — FULL 0x1 — NOTFULL	RO	0x1
0	activity	I2C Activity Status. Reset value: 0x0	RO	0x0

ic_txflr

```
Name: I2C Transmit FIFO Level Register

This register contains the number of valid data
entries in the transmit FIFO buffer. It is cleared
whenever:
- The I2C is disabled
- There is a transmit abort that is, TX_ABRT bit is
  set in the IC_RAW_INTR_STAT register
- The slave bulk transmit mode is aborted
The register increments whenever data is placed into
the transmit FIFO and decrements when data is
taken from the transmit FIFO.
```

Module Instance	Base Address	Register Address
i_i2c_0_i2c	0xFFC02200	0xFFC02274
i_i2c_1_i2c	0xFFC02300	0xFFC02374

Offset: 0x74

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									txflr RO 0x0						

ic_txflr Fields

Bit	Name	Description	Access	Reset
6:0	txflr	Transmit FIFO Level. Contains the number of valid data entries in the transmit FIFO. Reset value: 0x0	RO	0x0

ic_rxflr

```
I2C Receive FIFO Level Register

This register contains the number of valid data
entries in the receive FIFO buffer. It is cleared
whenever:
- The I2C is disabled
- Whenever there is a transmit abort caused by any
  of the events tracked in IC_TX_ABRT_SOURCE
The register increments whenever data is placed into
the receive FIFO and decrements when data is
taken from the receive FIFO.
```

Module Instance	Base Address	Register Address
i_i2c_0_i2c	0xFFC02200	0xFFC02278
i_i2c_1_i2c	0xFFC02300	0xFFC02378

Offset: 0x78

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									rxflr RO 0x0						

ic_rxflr Fields

Bit	Name	Description	Access	Reset
6:0	rxflr	Receive FIFO Level. Contains the number of valid data entries in the receive FIFO. Reset value: 0x0	RO	0x0

ic_sda_hold

```
Name: I2C SDA Hold Time Length Register

The bits [15:0] of this register are used to control the hold time of
SDA during
transmit in both slave and master mode (after SCL goes from HIGH to
LOW).
The bits [23:16] of this register are used to extend the SDA
transition (if any)
whenever SCL is HIGH in the receiver in either master or slave mode.
The values in this register are in units of ic_clk period.
This register controls the amount of time delay.
The relevant I2C requirement is thd:DAT as detailed in the I2C
Bus Specification.
```

Module Instance	Base Address	Register Address
i_i2c_0_i2c	0xFFC02200	0xFFC0227C
i_i2c_1_i2c	0xFFC02300	0xFFC0237C

Offset: 0x7C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								ic_sda_rx_hold RW 0x0							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ic_sda_tx_hold RW 0x1															

ic_sda_hold Fields

Bit	Name	Description	Access	Reset
23:16	ic_sda_rx_hold	Sets the required SDA hold time in units of ic_clk period, when the I2C module acts as a receiver.	RW	0x0

Bit	Name	Description	Access	Reset
15:0	ic_sda_tx_hold	Sets the required SDA hold time in units of ic_clk period, when the I2C module acts as a transmitter.	RW	0x1

ic_tx_abrt_source

I2C Transmit Abort Source Register

This register has 32 bits that indicate the source of the TX_ABRT bit. Except for Bit 9, this register is cleared whenever the IC_CLR_TX_ABRT register or the IC_CLR_INTR register is read. To clear Bit 9, the source of the ABRT_SBYTE_NORSTRT must be fixed first; RESTART must be enabled (IC_CON[5]=1), the SPECIAL bit must be cleared (IC_TAR[11]), or the GC_OR_START bit must be cleared (IC_TAR[10]). Once the source of the ABRT_SBYTE_NORSTRT is fixed, then this bit can be cleared in the same manner as other bits in this register. If the source of the ABRT_SBYTE_NORSTRT is not fixed before attempting to clear this bit, Bit 9 clears for one cycle and is then re-asserted.

Module Instance	Base Address	Register Address
i_i2c_0_i2c	0xFFC02200	0xFFC02280
i_i2c_1_i2c	0xFFC02300	0xFFC02380

Offset: 0x80

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
tx_flush_cnt RO 0x0									rsvd_ic_tx_abrt_source_22to17 RO 0x0						abrt_user_abrt RO 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
abrt_slvrd_intx RO 0x0	abrt_slv_arblost RO 0x0	abrt_slvflush_txfifo RO 0x0	arb_lost RO 0x0	abrt_master_dis RO 0x0	abrt_10b_rd_norstrt RO 0x0	abrt_sbyte_norstrt RO 0x0	abrt_hs_norstrt RO 0x0	abrt_sbyte_ackdet RO 0x0	abrt_hs_ackdet RO 0x0	abrt_gcall_read RO 0x0	abrt_gcall_noack RO 0x0	abrt_txdata_noack RO 0x0	abrt_10addr2_noack RO 0x0	abrt_10addr1_noack RO 0x0	abrt_7b_addr_noack RO 0x0

ic_tx_abrt_source Fields

Bit	Name	Description	Access	Reset
31:23	tx_flush_cnt	This field indicates the number of Tx FIFO Data Commands which are flushed due to TX_ABRT interrupt. It is cleared whenever I2C is disabled. Reset value: 0x0 Role of the I2C module: Master-Transmitter or Slave-Transmitter	RO	0x0
22:17	rsvd_ic_tx_abrt_source_22to17	Reserved Reset value: 0x0	RO	0x0
16	abrt_user_abrt	This is a master-mode-only bit. Master has detected the transfer abort (IC_ENABLE[1]) Reset value: 0x0 Role of the I2C module: Master-Transmitter	RO	0x0
15	abrt_slvrd_intx	1: When the processor side responds to a slave mode request for data to be transmitted to a remote master and user writes a 1 in CMD (bit 8) of IC_DATA_CMD register. Reset value: 0x0 Role of I2C module: Slave-Transmitter	RO	0x0

Bit	Name	Description	Access	Reset
14	abrt_slv_arblost	1: Slave lost the bus while transmitting data to a remote master. IC_TX_ABRT_SOURCE[12] is set at the same time. Note: Even though the slave never 'owns' the bus, something could go wrong on the bus. This is a fail safe check. For instance, during a data transmission at the low-to-high transition of SCL, if what is on the data bus is not what is supposed to be transmitted, then the I2C module no longer own the bus. Reset value: 0x0 Role of the I2C module: Slave-Transmitter	RO	0x0
13	abrt_slvflush_txfifo	1: Slave has received a read command and some data exists in the TX FIFO so the slave issues a TX_ABRT interrupt to flush old data in TX FIFO. Reset value: 0x0 Role of the I2C module: Slave-Transmitter	RO	0x0
12	arb_lost	1: Master has lost arbitration, or if IC_TX_ABRT_SOURCE[14] is also set, then the slave transmitter has lost arbitration. Reset value: 0x0 Role of the I2C module: Master-Transmitter or Slave-Transmitter	RO	0x0
11	abrt_master_dis	1: User tries to initiate a Master operation with the Master mode disabled. Reset value: 0x0 Role of the I2C module: Master-Transmitter or Master-Receiver	RO	0x0

Bit	Name	Description	Access	Reset
10	abrt_10b_rd_norstrt	1: The restart is disabled (IC_RESTART_EN bit (IC_CON[5]) =0) and the master sends a read command in 10-bit addressing mode. Reset value: 0x0 Role of the I2C module: Master-Receiver	RO	0x0
9	abrt_sbyte_norstrt	To clear Bit 9, the source of the ABRT_SBYTE_NORSTRT must be fixed first; restart must be enabled (IC_CON[5]=1), the SPECIAL bit must be cleared (IC_TAR[11]), or the GC_OR_START bit must be cleared (IC_TAR[10]). Once the source of the ABRT_SBYTE_NORSTRT is fixed, then this bit can be cleared in the same manner as other bits in this register. If the source of the ABRT_SBYTE_NORSTRT is not fixed before attempting to clear this bit, bit 9 clears for one cycle and then gets reasserted. 1: The restart is disabled (IC_RESTART_EN bit (IC_CON[5]) =0) and the user is trying to send a START Byte. Reset value: 0x0 Role of the I2C module: Master	RO	0x0
8	abrt_hs_norstrt	1: The restart is disabled (IC_RESTART_EN bit (IC_CON[5]) =0) and the user is trying to use the master to transfer data in High Speed mode. Reset value: 0x0 Role of the I2C module: Master-Transmitter or Master-Receiver	RO	0x0

Bit	Name	Description	Access	Reset
7	abrt_sbyte_ackdet	1: Master has sent a START Byte and the START Byte was acknowledged (wrong behavior). Reset value: 0x0 Role of the I2C module: Master	RO	0x0
6	abrt_hs_ackdet	1: Master is in High Speed mode and the High Speed Master code was acknowledged (wrong behavior). Reset value: 0x0 Role of the I2C module: Master	RO	0x0
5	abrt_gcall_read	1: DW_apb_i2c in master mode sent a General Call but the user programmed the byte following the General Call to be a read from the bus (IC_DATA_CMD[9] is set to 1). Reset value: 0x0 Role of the I2C module: Master-Transmitter	RO	0x0
4	abrt_gcall_noack	1: DW_apb_i2c in master mode sent a General Call and no slave on the bus acknowledged the General Call. Reset value: 0x0 Role of the I2C module: Master-Transmitter	RO	0x0
3	abrt_txdata_noack	1: This is a master-mode only bit. Master has received an acknowledgement for the address, but when it sent data byte(s) following the address, it did not receive an acknowledge from the remote slave(s). Reset value: 0x0 Role of the I2C module: Master-Transmitter	RO	0x0

Bit	Name	Description	Access	Reset
2	abrt_10addr2_noack	1: Master is in 10-bit address mode and the second address byte of the 10-bit address was not acknowledged by any slave. Reset value: 0x0 Role of the I2C module: Master-Transmitter or Master-Receiver	RO	0x0
1	abrt_10addr1_noack	1: Master is in 10-bit address mode and the first 10-bit address byte was not acknowledged by any slave. Reset value: 0x0 Role of the I2C module: Master-Transmitter or Master-Receiver	RO	0x0
0	abrt_7b_addr_noack	1: Master is in 7-bit addressing mode and the address sent was not acknowledged by any slave. Reset value: 0x0 Role of the I2C module: Master-Transmitter or Master-Receiver	RO	0x0

ic_slv_data_nack_only

Generate Slave Data NACK Register

The register is used to generate a NACK for the data part of a transfer when the I2C is acting as a slave-receiver.

Module Instance	Base Address	Register Address
i_i2c_0_i2c	0xFFC02200	0xFFC02284
i_i2c_1_i2c	0xFFC02300	0xFFC02384

Offset: 0x84

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															nack RW 0x0

ic_slv_data_nack_only Fields

Bit	Name	Description	Access	Reset
0	nack	Generate NACK. This NACK generation only occurs when the I2C module is a slave-receiver. If this register is set to a value of 1, it can only generate a NACK after a data byte is received; hence, the data transfer is aborted and the data received is not pushed to the receive buffer. When the register is set to a value of 0, it generates NACK/ACK, depending on normal criteria. 1 = generate NACK after data byte received 0 = generate NACK/ACK normally Reset value: 0x0 **Value** **Description** 0x0 NORM 0x1 AFTERDBYTE	RW	0x0

ic_dma_cr

Name: DMA Control Register

The register is used to enable the DMA Controller interface operation. There is a separate bit for transmit and receive. This can be programmed

regardless of the state of IC_ENABLE.

Module Instance	Base Address	Register Address
i_i2c_0_i2c	0xFFC02200	0xFFC02288
i_i2c_1_i2c	0xFFC02300	0xFFC02388

Offset: 0x88

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
rsvd_ic_dma_cr_31to2 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rsvd_ic_dma_cr_31to2 RO 0x0														tdmae RW 0x0	rdmae RW 0x0

ic_dma_cr Fields

Bit	Name	Description	Access	Reset
31:2	rsvd_ic_dma_cr_31to2	Reserved bits [31:1] - Read Only	RO	0x0

1	tdmae	Transmit DMA Enable.
		//This bit enables/disables the transmit FIFO DMA channel.
		0 = Transmit DMA disabled
		1 = Transmit DMA enabled
		Reset value: 0x0
		Value **Description**
		0x0 DISABLE
		0x1 ENABLE
			RW	0x0

Bit	Name	Description	Access	Reset
0	rdmae	Receive DMA Enable. This bit enables/disables the receive FIFO DMA channel. 0 = Receive DMA disabled 1 = Receive DMA enabled Reset value: 0x0 Value: 0x0 DISABLE; 0x1 ENABLE	RW	0x0

ic_dma_tdlr

DMA Transmit Data Level Register

Module Instance	Base Address	Register Address
i_i2c_0_i2c	0xFFC02200	0xFFC0228C
i_i2c_1_i2c	0xFFC02300	0xFFC0238C

Offset: 0x8C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										dmatdl RW 0x0					

ic_dma_tdlr Fields

Bit	Name	Description	Access	Reset
5:0	dmatdl	Transmit Data Level. This bit field controls the level at which a DMA request is made by the transmit logic. It is equal to the watermark level; that is, the dma_tx_req signal is generated when the number of valid data entries in the transmit FIFO is equal to or below this field value, and TDMAE = 1. Reset value: 0x0	RW	0x0

ic_dma_rdlr

I2C Receive Data Level Register

Module Instance	Base Address	Register Address
i_i2c_0_i2c	0xFFC02200	0xFFC02290
i_i2c_1_i2c	0xFFC02300	0xFFC02390

Offset: 0x90

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										dmardl RW 0x0					

ic_dma_rdlr Fields

Bit	Name	Description	Access	Reset
5:0	dmardl	Receive Data Level. This bit field controls the level at which a DMA request is made by the receive logic. The watermark level = DMARDL+1; that is, dma_rx_req is generated when the number of valid data entries in the receive FIFO is equal to or more than this field value + 1, and RDMAE =1. For instance, when DMARDL is 0, then dma_rx_req is asserted when 1 or more data entries are present in the receive FIFO. Reset value: 0x0	RW	0x0

ic_sda_setup

I2C SDA Setup Register

This register controls the amount of time delay
(in terms of number of l4_sp_clk clock periods) introduced
in the rising edge of SCL, relative to SDA changing, when
the I2C module services a read request in a slave-transmitter
operation.
The relevant I2C requirement is tSU:DAT (note 4) as detailed in the
I2C Bus Specification.

Module Instance	Base Address	Register Address
i_i2c_0_i2c	0xFFC02200	0xFFC02294
i_i2c_1_i2c	0xFFC02300	0xFFC02394

Offset: 0x94

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								sda_setup RW 0x64							

ic_sda_setup Fields

Bit	Name	Description	Access	Reset
7:0	sda_setup	SDA Setup. It is recommended that if the required delay is 1000ns, then for an l4_sp_clk frequency of 10 MHz, IC_SDA_SETUP should be programmed to a value of 11. Default Reset value: 0x64	RW	0x64

ic_ack_general_call

I2C ACK General Call Register

The register controls whether the I2C module responds with a ACK or NACK when it receives an I2C General Call address.
Note :This register is applicable only when the I2C module is in slave mode

Module Instance	Base Address	Register Address
i_i2c_0_i2c	0xFFC02200	0xFFC02298
i_i2c_1_i2c	0xFFC02300	0xFFC02398

Offset: 0x98

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
rsvd_ic_ack_gen_31to1 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rsvd_ic_ack_gen_31to1 RO 0x0															ack_gen_call RW 0x1

ic_ack_general_call Fields

Bit	Name	Description	Access	Reset
31:1	rsvd_ic_ack_gen_31to1	Reserved bits [31:1] - Read Only	RO	0x0
0	ack_gen_call	ACK General Call. When set to 1, the I2C module responds with a ACK (by asserting ic_data_oe) when it receives a General Call. Otherwise, the I2C module responds with a NACK (by negating ic_data_oe). Default Reset value: 0x1 **Value** **Description** 0x0 NACK 0x1 ACK	RW	0x1

ic_enable_status

I2C Enable Status Register

The register is used to report the I2C hardware status when the IC_ENABLE[0] register is set from 1 to 0; that is, when the I2C module is disabled.
If IC_ENABLE[0] has been set to 1, bits 2:1 are forced to 0, and bit 0 is forced to 1.
If IC_ENABLE[0] has been set to 0, bits 2:1 is only be valid as soon as bit 0 is read as '0'.
Note
When IC_ENABLE[0] has been written with '0'a delay occurs for bit 0 to be read as '0' because disabling the I2C module depends on I2C bus activities.

Module Instance	Base Address	Register Address
i_i2c_0_i2c	0xFFC02200	0xFFC0229C
i_i2c_1_i2c	0xFFC02300	0xFFC0239C

Offset: 0x9C

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													slv_rx_data_lost RO 0x0	slv_disabled_while_busy RO 0x0	ic_en RO 0x0

ic_enable_status Fields

Bit	Name	Description	Access	Reset
2	slv_rx_data_lost	Slave Received Data Lost. This bit indicates if a Slave-Receiver operation has been aborted with at least one data byte received from an I2C transfer due to the setting bit 0 of IC_ENABLE from 1 to 0. When read as 1, the I2C module is deemed to have been actively engaged in an aborted I2C transfer (with matching address) and the data phase of the I2C transfer has been entered, even though a data byte has been responded with a NACK. NOTE: If the remote I2C master terminates the transfer with a STOP condition before the I2C module has a chance to NACK a transfer, and IC_ENABLE[0] has been set to 0, then this bit is also set to 1. When read as 0, the I2C module is deemed to have been disabled without being actively involved in the data phase of a Slave-Receiver transfer. NOTE: The CPU can safely read this bit when IC_EN (bit 0) is read as 0. Reset value: 0x0	RO	0x0

Bit	Name	Description	Access	Reset
1	slv_disabled_while_busy	Slave Disabled While Busy (Transmit, Receive). This bit indicates if a potential or active Slave operation has been aborted due to the setting bit 0 of the IC_ENABLE register from 1 to 0. This bit is set when the CPU writes a 0 to the IC_ENABLE register while: (a) The I2C is receiving the address byte of the Slave-Transmitter operation from a remote master; OR, (b) address and data bytes of the Slave-Receiver operation from a remote master. When read as 1, the I2C module is deemed to have forced a NACK during any part of an I2C transfer, irrespective of whether the I2C address matches the slave address set in the I2C (IC_SAR register) OR if the transfer is completed before IC_ENABLE is set to 0 but has not taken effect. NOTE: If the remote I2C master terminates the transfer with a STOP condition before the I2C module has a chance to NACK a transfer, and IC_ENABLE[0] has been set to 0, then this bit will also be set to 1. When read as 0, the I2C module is deemed to have been disabled when there is master activity, or when the I2C bus is idle. NOTE: The CPU can safely read this bit when IC_EN (bit 0) is read as 0. Reset value: 0x0	RO	0x0

Bit	Name	Description	Access	Reset
0	ic_en	ic_en Status. This bit always reflects the value driven on the output port ic_en. When read as 1, the I2C module is deemed to be in an enabled state. When read as 0, the I2C module is deemed completely inactive. NOTE: The CPU can safely read this bit anytime. When this bit is read as 0, the CPU can safely read SLV_RX_DATA_LOST (bit 2) and SLV_DISABLED_WHILE_BUSY (bit 1). Reset value: 0x0	RO	0x0

ic_comp_param_1

Component Parameter Register 1

Note
This is a constant read-only register that contains encoded information about the component's parameter settings.

Module Instance	Base Address	Register Address
i_i2c_0_i2c	0xFFC02200	0xFFC022F4
i_i2c_1_i2c	0xFFC02300	0xFFC023F4

Offset: 0xF4

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								tx_buffer_depth RO 0x3F							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rx_buffer_depth RO 0x3F								add_encoded_params RO 0x1	has_dma RO 0x1	intr_io RO 0x1	hc_count_values RO 0x0	max_speed_mode RO 0x2		apb_data_width RO 0x2	

ic_comp_param_1 Fields

Bit	Name	Description	Access	Reset
23:16	tx_buffer_depth	This field identifies the Tx Buffer Depth **Value** **Description** 0x40 FIFO64BYTES	RO	0x3F
15:8	rx_buffer_depth	This field identifies the Rx Buffer Depth **Value** **Description** 0x40 FIFO64BYTES	RO	0x3F
7	add_encoded_params	Reading 1 in this bit means that the capability of reading these encoded parameters via software has been included. Otherwise, the entire register is 0 regardless of the setting of any other parameters that are encoded in the bits. 0: False 1: True **Value** **Description** 0x1 ADDENCPARAMS	RO	0x1

Bit	Name	Description	Access	Reset
6	has_dma	This bit identifies if the DMA is present. Value: 0x1 — Description: PRESENT	RO	0x1
5	intr_io	This bit indicates if interrupt outputs are individual or combined into a single output. Value: 0x1 — Description: COMBINED	RO	0x1
4	hc_count_values	This field indicates if the CNT registers are read only or read/write. Value: 0x0 — Description: READWRITE	RO	0x0
3:2	max_speed_mode	This field indicates the maximum I2C bus speed. Value: 0x2 — Description: FAST	RO	0x2
1:0	apb_data_width	The field identifies the width of the APB bus. Value: 0x2 — Description: WIDTH32BITS	RO	0x2

ic_comp_version

I2C Component Version Register

Module Instance	Base Address	Register Address
i_i2c_0_i2c	0xFFC02200	0xFFC022F8
i_i2c_1_i2c	0xFFC02300	0xFFC023F8

Offset: 0xF8

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ic_comp_version RO 0x3132312A															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ic_comp_version RO 0x3132312A															

ic_comp_version Fields

Bit	Name	Description	Access	Reset
31:0	ic_comp_version	**Value** **Description** 0x3132312a VER_1_21A	RO	0x3132312A

ic_comp_type

I2C Component Type Register

Module Instance	Base Address	Register Address
i_i2c_0_i2c	0xFFC02200	0xFFC022FC
i_i2c_1_i2c	0xFFC02300	0xFFC023FC

Offset: 0xFC

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ic_comp_type RO 0x44570140															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ic_comp_type RO 0x44570140															

ic_comp_type Fields

Bit	Name	Description	Access	Reset
31:0	ic_comp_type	Designware Component Type number = 0x44_57_01_40. This assigned unique hex value is constant and is derived from the two ASCII letters 'DW' followed by a 16-bit unsigned number.	RO	0x44570140

ic_fs_spklen

I2C SS, FS or FM+ spike suppression limit

This register is used to store the duration, measured in l4_sp_clk cycles, of the longest spike that is filtered out by the spike suppression logic when the component is operating in SS, FS or FM+ modes. The relevant I2C requirement is tSP (table 4) as detailed in the I2C Bus Specification. This register must be programmed with a minimum value of 1.

Module Instance	Base Address	Register Address
i_i2c_0_i2c	0xFFC02200	0xFFC022A0
i_i2c_1_i2c	0xFFC02300	0xFFC023A0

Offset: 0xA0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								ic_fs_spklen RW 0x2							

ic_fs_spklen Fields

Bit	Name	Description	Access	Reset
7:0	ic_fs_spklen	This register must be set before any I2C bus transaction can take place to ensure stable operation. This register sets the duration, measured in l4_sp_clk cycles, of the longest spike in the SCL or SDA lines that will be filtered out by the spike suppression logic. This register can be written only when the I2C interface is disabled which corresponds to the IC_ENABLE[0] register being set to 0. Writes at other times have no effect. The minimum valid value is 1; hardware prevents values less than this being written, and if attempted results in 1 being set.	RW	0x2

ic_clr_restart_det

Clear RESTART_DET Interrupt Register

Module Instance	Base Address	Register Address
i_i2c_0_i2c	0xFFC02200	0xFFC022A8
i_i2c_1_i2c	0xFFC02300	0xFFC023A8

Offset: 0xA8

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															clr_restart_det RO 0x0

ic_clr_restart_det Fields

Bit	Name	Description	Access	Reset
0	clr_restart_det	Read this register to clear the RESTART_DET interrupt (bit 12) of IC_RAW_INTR_STAT register. Reset value: 0x0	RO	0x0

i2c_emac Address Map

Module Instance	Base Address	End Address
i_i2c_emac_0_i2c	0xFFC02400	0xFFC024FF
i_i2c_emac_1_i2c	0xFFC02500	0xFFC025FF
i_i2c_emac_2_i2c	0xFFC02600	0xFFC026FF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
ic_con on page 20-163	0x0	32	RW	0x7D	Name: I2C Control Register Size: 10 bits Address Offset: 0x00 Read/Write Access: If configuration parameter I2C_DYNAMIC_TAR_UPDATE = 0, all bits are Read/Write. If I2C_DYNAMIC_TAR_UPDATE = 1, bit 4 is Read-only. This register can be written only when the DW_apb_i2c is disabled, which corresponds to the IC_ENABLE[0] register being set to 0. Writes at other times have no effect.

Register	Offset	Width	Access	Reset Value	Description
ic_tar on page 20-168	0x4	32	RW	0x1055	Name: I2C Target Address Register Size: 12 bits or 13 bits; 13 bits only when I2C_DYNAMIC_TAR_UPDATE = 1 Address Offset: 0x04 Read/Write Access: Read/Write If the configuration parameter I2C_DYNAMIC_TAR_UPDATE is set to 'No' (0), this register is 12 bits wide, and bits 31:12 are reserved. This register can be written to only when IC_ENABLE[0] is set to 0. However, if I2C_DYNAMIC_TAR_UPDATE = 1, then the register becomes 13 bits wide. All bits can be dynamically updated as long as any set of the following conditions are true: - DW_apb_i2c is NOT enabled (IC_ENABLE[0] is set to 0); or - DW_apb_i2c is enabled (IC_ENABLE[0]=1); AND DW_apb_i2c is NOT engaged in any Master (tx, rx) operation (IC_STATUS[5]=0); AND DW_apb_i2c is enabled to operate in Master mode (IC_CON[0]=1); AND there are NO entries in the TX FIFO (IC_STATUS[2]=1)

Register	Offset	Width	Access	Reset Value	Description
ic_sar on page 20-170	0x8	32	RW	0x55	Name: I2C Slave Address Register Size: 10 bits Address Offset: 0x08 Read/Write Access: Read/Write
ic_data_cmd on page 20-171	0x10	32	RW	0x0	Name: I2C Rx/Tx Data Buffer and Command Register; this is the register the CPU writes to when filling the TX FIFO and the CPU reads from when retrieving bytes from RX FIFO Size: When IC_EMPTYFIFO_HOLD_MASTER_EN=1 - 11 bits (writes), 8 bits (read) When IC_EMPTYFIFO_HOLD_MASTER_EN=0 - 9 bits (writes), 8 bits (read) Address Offset: 0x10 Read/Write Access: Read/Write NOTE: With nine bits required for writes, the DW_apb_i2c requires 16-bit data on the APB bus transfers when writing into the transmit FIFO. Eight-bit transfers remain for reads from the receive FIFO.
ic_ss_scl_hcnt on page 20-175	0x14	32	RW	0x190	Name: Standard Speed I2C Clock SCL High Count Register Size: 16 bits Address Offset: 0x14 Read/Write Access: Read/Write

Register	Offset	Width	Access	Reset Value	Description
ic_ss_scl_lcnt on page 20-176	0x18	32	RW	0x1D6	Name: Standard Speed I2C Clock SCL Low Count Register Size: 16 bits Address Offset: 0x18 Read/Write Access: Read/Write
ic_fs_scl_hcnt on page 20-177	0x1C	32	RW	0x3C	Name: Fast Mode or Fast Mode Plus I2C Clock SCL High Count Register Size: 16 bits Address Offset: 0x1c Read/Write Access: Read/Write
ic_fs_scl_lcnt on page 20-178	0x20	32	RW	0x82	Name: Fast Mode or Fast Mode Plus I2C Clock SCL Low Count Register Size: 16 bits Address Offset: 0x20 Read/Write Access: Read/Write
ic_intr_stat on page 20-179	0x2C	32	RO	0x0	Name: I2C Interrupt Status Register Size: 14 bits Address Offset: 0x2C Read/Write Access: Read Each bit in this register has a corresponding mask bit in the IC_INTR_MASK register. These bits are cleared by reading the matching interrupt clear register. The unmasked raw versions of these bits are available in the IC_RAW_INTR_STAT register.

Register	Offset	Width	Access	Reset Value	Description
ic_intr_mask on page 20-187	0x30	32	RW	0x8FF	Name: I2C Interrupt Mask Register Size: 14 bits Address Offset: 0x30 Read/Write Access: Read/Write However, if configuration parameter IC_SLV_RESTART_DET = 0, bit 13 is read only; if configuration parameter I2C_DYNAMIC_TAR_UPDATE = 0 or IC_EMPTYFIFO_HOLD_MASTER_EN = 0, bit 14 is read only. These bits mask their corresponding interrupt status bits. They are active high; a value of 0 prevents a bit from generating an interrupt.
ic_raw_intr_stat on page 20-189	0x34	32	RO	0x0	Name: I2C Raw Interrupt Status Register Size: 14 bits Address Offset: 0x34 Read/Write Access: Read Unlike the IC_INTR_STAT register, these bits are not masked so they always show the true status of the DW_apb_i2c.
ic_rx_tl on page 20-196	0x38	32	RW	0x0	Name: I2C Receive FIFO Threshold Register Size: 8bits Address Offset: 0x38 Read/Write Access: Read/Write

Register	Offset	Width	Access	Reset Value	Description
ic_tx_tl on page 20-197	0x3C	32	RW	0x0	Name: I2C Transmit FIFO Threshold Register Size: 8 bits Address Offset: 0x3c Read/Write Access: Read/Write
ic_clr_intr on page 20-198	0x40	32	RO	0x0	Name: Clear Combined and Individual Interrupt Register Size: 1 bit Address Offset: 0x40 Read/Write Access: Read
ic_clr_rx_under on page 20-198	0x44	32	RO	0x0	Name: Clear RX_UNDER Interrupt Register Size: 1 bit Address Offset: 0x44 Read/Write Access: Read
ic_clr_rx_over on page 20-199	0x48	32	RO	0x0	Name: Clear RX_OVER Interrupt Register Size: 1 bit Address Offset: 0x48 Read/Write Access: Read
ic_clr_tx_over on page 20-200	0x4C	32	RO	0x0	Name: Clear TX_OVER Interrupt Register Size: 1 bit Address Offset: 0x4c Read/Write Access: Read
ic_clr_rd_req on page 20-201	0x50	32	RO	0x0	Name: Clear RD_REQ Interrupt Register Size: 1 bit Address Offset: 0x50 Read/Write Access: Read
ic_clr_tx_abrt on page 20-202	0x54	32	RO	0x0	Name: Clear TX_ABRT Interrupt Register Size: 1 bit Address Offset: 0x54 Read/Write Access: Read
ic_clr_rx_done on page 20-203	0x58	32	RO	0x0	Name: Clear RX_DONE Interrupt Register Size: 1 bit Address Offset: 0x58 Read/Write Access: Read

Register	Offset	Width	Access	Reset Value	Description
ic_clr_activity on page 20-204	0x5C	32	RO	0x0	Name: Clear ACTIVITY Interrupt Register Size: 1 bit Address Offset: 0x5c Read/Write Access: Read
ic_clr_stop_det on page 20-205	0x60	32	RO	0x0	Name: Clear STOP_DET Interrupt Register Size: 1 bit Address Offset: 0x60 Read/Write Access: Read
ic_clr_start_det on page 20-206	0x64	32	RO	0x0	Name: Clear START_DET Interrupt Register Size: 1 bit Address Offset: 0x64 Read/Write Access: Read
ic_clr_gen_call on page 20-207	0x68	32	RO	0x0	Name: Clear GEN_CALL Interrupt Register Size: 1 bit Address Offset: 0x68 Read/Write Access: Read
ic_enable on page 20-208	0x6C	32	RW	0x0	Name: I2C Enable Register Size: 2 bits Address Offset: 0x6c Read/Write Access: Read/Write

Register	Offset	Width	Access	Reset Value	Description
ic_status on page 20-210	0x70	32	RO	0x6	Name: I2C Status Register Size: 7 bits Address Offset: 0x70 Read/Write Access: Read This is a read-only register used to indicate the current transfer status and FIFO status. The status register may be read at any time. None of the bits in this register request an interrupt. When the I2C is disabled by writing 0 in bit 0 of the IC_ENABLE register: - Bits 1 and 2 are set to 1 - Bits 3 and 4 are set to 0 When the master or slave state machines goes to idle and ic_en=0: - Bits 5 and 6 are set to 0

Register	Offset	Width	Access	Reset Value	Description
ic_txflr on page 20-214	0x74	32	RO	0x0	Name: I2C Transmit FIFO Level Register Size: TX_ABW + 1 Address Offset: 0x74 Read/Write Access: Read This register contains the number of valid data entries in the transmit FIFO buffer. It is cleared whenever: - The I2C is disabled - There is a transmit abort that is, TX_ABRT bit is set in the IC_RAW_INTR_STAT register - The slave bulk transmit mode is aborted The register increments whenever data is placed into the transmit FIFO and decrements when data is taken from the transmit FIFO.

Register	Offset	Width	Access	Reset Value	Description
ic_rxflr on page 20-215	0x78	32	RO	0x0	Name: I2C Receive FIFO Level Register Size: RX_ABW + 1 Address Offset: 0x78 Read/Write Access: Read This register contains the number of valid data entries in the receive FIFO buffer. It is cleared whenever: - The I2C is disabled - Whenever there is a transmit abort caused by any of the events tracked in IC_TX_ABRT_SOURCE The register increments whenever data is placed into the receive FIFO and decrements when data is taken from the receive FIFO.

Register	Offset	Width	Access	Reset Value	Description
ic_sda_hold on page 20-216	0x7C	32	RW	0x1	Name: I2C SDA Hold Time Length Register Size: 24 bits Address Offset: 0x7c Read/Write Access: Read/Write The bits [15:0] of this register are used to control the hold time of SDA during transmit in both slave and master mode (after SCL goes from HIGH to LOW). The bits [23:16] of this register are used to extend the SDA transition (if any) whenever SCL is HIGH in the receiver in either master or slave mode. The values in this register are in units of ic_clk period. This register controls the amount of time delay. The relevant I2C requirement is thd:DAT as detailed in the I2C Bus Specification.

Register	Offset	Width	Access	Reset Value	Description
ic_tx_abrt_source on page 20-217	0x80	32	RO	0x0	Name: I2C Transmit Abort Source Register Size: 32 bits Address Offset: 0x80 Read/Write Access: Read This register has 32 bits that indicate the source of the TX_ABRT bit. Except for Bit 9, this register is cleared whenever the IC_CLR_TX_ABRT register or the IC_CLR_INTR register is read. To clear Bit 9, the source of the ABRT_SBYTE_NORSTRT must be fixed first; RESTART must be enabled (IC_CON[5]=1), the SPECIAL bit must be cleared (IC_TAR[11]), or the GC_OR_START bit must be cleared (IC_TAR[10]). Once the source of the ABRT_SBYTE_NORSTRT is fixed, then this bit can be cleared in the same manner as other bits in this register. If the source of the ABRT_SBYTE_NORSTRT is not fixed before attempting to clear this bit, Bit 9 clears for one cycle and is then re-asserted.

Register	Offset	Width	Access	Reset Value	Description
ic_slv_data_nack_only on page 20-222	0x84	32	RW	0x0	Name: Generate Slave Data NACK Register Size: 1 bit Address Offset: 0x84 Read/Write Access: Read/Write The register is used to generate a NACK for the data part of a transfer when DW_apb_i2c is acting as a slave-receiver. This register only exists when the IC_SLV_DATA_NACK_ONLY parameter is set to 1. When this parameter disabled, this register does not exist and writing to the register's address has no effect.

Register	Offset	Width	Access	Reset Value	Description
ic_dma_cr on page 20-223	0x88	32	RW	0x0	Name: DMA Control Register Size: 2 bits Address Offset: 0x88 Read/Write Access: Read/Write This register is only valid when DW_apb_i2c is configured with a set of DMA Controller interface signals (IC_HAS_DMA = 1). When DW_apb_i2c is not configured for DMA operation, this register does not exist and writing to the register's address has no effect and reading from this register address will return zero. The register is used to enable the DMA Controller interface operation. There is a separate bit for transmit and receive. This can be programmed regardless of the state of IC_ENABLE.

Register	Offset	Width	Access	Reset Value	Description
ic_dma_tdlr on page 20-225	0x8C	32	RW	0x0	Name: DMA Transmit Data Level Register Size: log2(IC_TX_BUFFER_DEPTH) bits Address Offset: 0x8c Read/Write Access: Read/Write This register is only valid when the DW_apb_i2c is configured with a set of DMA interface signals (IC_HAS_DMA = 1). When DW_apb_i2c is not configured for DMA operation, this register does not exist; writing to its address has no effect; reading from its address returns zero.
ic_dma_rdlr on page 20-226	0x90	32	RW	0x0	Name: I2C Receive Data Level Register Size: log2(IC_RX_BUFFER_DEPTH) bits Address Offset: 0x90 Read/Write Access: Read/Write This register is only valid when DW_apb_i2c is configured with a set of DMA interface signals (IC_HAS_DMA = 1). When DW_apb_i2c is not configured for DMA operation, this register does not exist; writing to its address has no effect; reading from its address returns zero.

Register	Offset	Width	Access	Reset Value	Description
ic_sda_setup on page 20-227	0x94	32	RW	0x64	Name: I2C SDA Setup Register Size: 8 bits Address Offset: 0x94 Read/Write Access: Read/Write This register controls the amount of time delay (in terms of number of ic_clk clock periods) introduced in the rising edge of SCL, relative to SDA changing, when DW_apb_i2c services a read request in a slave-transmitter operation. The relevant I2C requirement is tSU:DAT (note 4) as detailed in the I2C Bus Specification.
ic_ack_general_call on page 20-228	0x98	32	RW	0x1	Name: I2C ACK General Call Register Size: 1 bit Address Offset: 0x98 Read/Write Access: Read/Write The register controls whether DW_apb_i2c responds with a ACK or NACK when it receives an I2C General Call address. Note :This register is applicable only when the DW_apb_i2c is in slave mode

Register	Offset	Width	Access	Reset Value	Description
ic_enable_status on page 20-229	0x9C	32	RO	0x0	Name: I2C Enable Status Register Size: 3 bits Address Offset: 0x9C Read/Write Access: Read The register is used to report the DW_apb_i2c hardware status when the IC_ENABLE[0] register is set from 1 to 0; that is, when DW_apb_i2c is disabled. If IC_ENABLE[0] has been set to 1, bits 2:1 are forced to 0, and bit 0 is forced to 1. If IC_ENABLE[0] has been set to 0, bits 2:1 is only be valid as soon as bit 0 is read as '0'. Note When IC_ENABLE[0] has been written with '0'a delay occurs for bit 0 to be read as '0' because disabling the DW_apb_i2c depends on I2C bus activities.
ic_comp_param_1 on page 20-233	0xF4	32	RO	0x3F3FEA	Name: Component Parameter Register 1 Size: 32 bits Address Offset: 0xf4 Read/Write Access: Read Note This is a constant read-only register that contains encoded information about the component's parameter settings. The reset value depends on coreConsul-tant parameter(s).

Register	Offset	Width	Access	Reset Value	Description
ic_comp_version on page 20-235	0xF8	32	RO	0x3132312A	Name: I2C Component Version Register Size: 32 bits Address Offset: 0xf8 Read/Write Access: Read
ic_comp_type on page 20-236	0xFC	32	RO	0x44570140	Name: I2C Component Type Register Size: 32 bits Address Offset: 0xfc Read/Write Access: Read
ic_fs_spklen on page 20-237	0xA0	32	RW	0x2	Name: I2C SS, FS or FM + spike suppression limit Size: 8 bits Address: 0xA0 Read/Write Access: Read/Write This register is used to store the duration, measured in ic_clk cycles, of the longest spike that is filtered out by the spike suppres-sion logic w hen the component is operating in SS, FS or FM+ modes. The relevant I2C requirement is tSP (table 4) as detailed in the I2C Bus Specification. This register must be programmed with a minimum value of 1.
ic_clr_restart_det on page 20-238	0xA8	32	RO	0x0	Name: Clear RESTART_ DET Interrupt Register Size: 1 bit Address Offset: 0xA8 Read/Write Access: Read

i2c_emac Summary

Module Instance	Base Address
i_i2c_emac_0_i2c	0xFFC02400
i_i2c_emac_1_i2c	0xFFC02500
i_i2c_emac_2_i2c	0xFFC02600

Register Address Offset	Bit Fields															
i_i2c_emac_0_i2c																
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	rsvd_ic_con_31to10 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ic_con 0x0	rsvd_ic_con_31to10 RO 0x0						rx_fifo_full_hld_ctrl RW 0x0	tx_empty_ctrl RW 0x0	stop_det_ifaddressed RW 0x0	ic_slave_disable RW 0x1	ic_restart_en RW 0x1	ic_10bitaddr_master RO 0x1	ic_10bitaddr_slave RW 0x1	speed RW 0x2		master_mode RW 0x1
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ic_tar 0x4	Reserved			ic_10bitaddr_master RW 0x1	special RW 0x0	gc_or_start RW 0x0	ic_tar RW 0x55									
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ic_sar 0x8	Reserved						ic_sar RW 0x55									

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
ic_data_cmd 0x10	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved					restart WO 0x0	stop WO 0x0	cmd WO 0x0	dat RW 0x0							
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
ic_ss_scl_hcnt 0x14	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ic_ss_scl_hcnt RW 0x190															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
ic_ss_scl_lcnt 0x18	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ic_ss_scl_lcnt RW 0x1D6															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
ic_fs_scl_hcnt 0x1C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ic_fs_scl_hcnt RW 0x3C															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
ic_fs_scl_lcnt 0x20	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ic_fs_scl_lcnt RW 0x82															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
ic_intr_stat 0x2C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		r_master_on_hold RO 0x0	r_restart_det RO 0x0	r_gen_call RO 0x0	r_start_det RO 0x0	r_stop_det RO 0x0	r_activity RO 0x0	r_rx_done RO 0x0	r_tx_abrt RO 0x0	r_rd_req RO 0x0	r_tx_empty RO 0x0	r_tx_over RO 0x0	r_rx_full RO 0x0	r_rx_over RO 0x0	r_rx_under RO 0x0

Register Address Offset	Bit Fields															
ic_intr_mask 0x30	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		m_master_on_hold RW 0x0	m_restart_det RW 0x0	m_gen_call RW 0x1	m_start_det RW 0x0	m_stop_det RW 0x0	m_activity RW 0x0	m_rx_done RW 0x1	m_tx_abrt RW 0x1	m_rd_req RW 0x1	m_tx_empty RW 0x1	m_tx_over RW 0x1	m_rx_full RW 0x1	m_rx_over RW 0x1	m_rx_under RW 0x1
ic_raw_intr_stat 0x34	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		master_on_hold RO 0x0	restart_det RO 0x0	gen_call RO 0x0	start_det RO 0x0	stop_det RO 0x0	activity RO 0x0	rx_done RO 0x0	tx_abrt RO 0x0	rd_req RO 0x0	tx_empty RO 0x0	tx_over RO 0x0	rx_full RO 0x0	rx_over RO 0x0	rx_under RO 0x0
ic_rx_tl 0x38	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								rx_tl RW 0x0							
ic_tx_tl 0x3C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								tx_tl RW 0x0							
ic_clr_intr 0x40	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															clr_intr RO 0x0
ic_clr_rx_under 0x44	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															clr_rx_under RO 0x0

Register Address Offset	Bit Fields															
ic_clr_rx_over 0x48	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															clr_rx_over RO 0x0
ic_clr_tx_over 0x4C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															clr_tx_over RO 0x0
ic_clr_rd_req 0x50	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															clr_rd_req RO 0x0
ic_clr_tx_abrt 0x54	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															clr_tx_abort RO 0x0
ic_clr_rx_done 0x58	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															clr_rx_done RO 0x0
ic_clr_activity 0x5C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															clr_activity RO 0x0

Register Address Offset	Bit Fields															
ic_clr_stop_det 0x60	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															clr_stop_det RO 0x0
ic_clr_start_det 0x64	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															clr_start_det RO 0x0
ic_clr_gen_call 0x68	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															clr_gen_call RO 0x0
ic_enable 0x6C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														txabort RW 0x0	enable RW 0x0
ic_status 0x70	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved									slv_activity RO 0x0	mst_activity RO 0x0	rff RO 0x0	rfne RO 0x0	tfe RO 0x1	tfnf RO 0x1	activity RO 0x0
ic_txflr 0x74	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved									txflr RO 0x0						

Register Address Offset	Bit Fields															
ic_rxflr 0x78	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved									rxflr RO 0x0						
ic_sda_hold 0x7C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved								ic_sda_rx_hold RW 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ic_sda_tx_hold RW 0x1															
ic_tx_abrt_source 0x80	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	tx_flush_cnt RO 0x0									rsvd_ic_tx_abrt_source_22to17 RO 0x0						abrt_user_abrt RO 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	abrt_slvrd_intx RO 0x0	abrt_slv_arblost RO 0x0	abrt_slvflush_txfifo RO 0x0	arb_lost RO 0x0	abrt_master_dis RO 0x0	abrt_10b_rd_norstrt RO 0x0	abrt_sbyte_norstrt RO 0x0	abrt_hs_norstrt RO 0x0	abrt_sbyte_ackdet RO 0x0	abrt_hs_ackdet RO 0x0	abrt_gcall_read RO 0x0	abrt_gcall_noack RO 0x0	abrt_txdata_noack RO 0x0	abrt_10addr2_noack RO 0x0	abrt_10addr1_noack RO 0x0	abrt_7b_addr_noack RO 0x0
ic_slv_data_nack_only 0x84	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															nack RW 0x0
ic_dma_cr 0x88	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	rsvd_ic_dma_cr_31to2 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_ic_dma_cr_31to2 RO 0x0														tdmae RW 0x0	rdmae RW 0x0

Register Address Offset	Bit Fields															
ic_dma_tdlr 0x8C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved										dmatdl RW 0x0					
ic_dma_rdlr 0x90	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved										dmardl RW 0x0					
ic_sda_setup 0x94	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								sda_setup RW 0x64							
ic_ack_general_call 0x98	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	rsvd_ic_ack_gen_31to1 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_ic_ack_gen_31to1 RO 0x0															ack_gen_call RW 0x1
ic_enable_status 0x9C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													slv_rx_data_lost RO 0x0	slv_disabled_while_busy RO 0x0	ic_en RO 0x0

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved								tx_buffer_depth RO 0x3F							
ic_comp_param_1 0xF4	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rx_buffer_depth RO 0x3F								add_encoded_params RO 0x1	has_dma RO 0x1	intr_io RO 0x1	hc_count_values RO 0x0	max_speed_mode RO 0x2		apb_data_width RO 0x2	
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ic_comp_version RO 0x3132312A															
ic_comp_version 0xF8	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ic_comp_version RO 0x3132312A															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ic_comp_type RO 0x44570140															
ic_comp_type 0xFC	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ic_comp_type RO 0x44570140															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
ic_fs_spklen 0xA0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								ic_fs_spklen RW 0x2							
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
ic_clr_restart_det 0xA8	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															clr_restart_det RO 0x0
i_i2c_emac_1_i2c																

Register Address Offset	Bit Fields															
ic_con 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	rsvd_ic_con_31to10 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_ic_con_31to10 RO 0x0						rx_fifo_full_hld_ctrl RW 0x0	tx_empty_ctrl RW 0x0	stop_det_ifaddressed RW 0x0	ic_slave_disable RW 0x1	ic_restart_en RW 0x1	ic_10bitaddr_master RO 0x1	ic_10bitaddr_slave RW 0x1	speed RW 0x2		master_mode RW 0x1
ic_tar 0x4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved			ic_10bitaddr_master RW 0x1	special RW 0x0	gc_or_start RW 0x0	ic_tar RW 0x55									
ic_sar 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						ic_sar RW 0x55									
ic_data_cmd 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved					restart WO 0x0	stop WO 0x0	cmd WO 0x0	dat RW 0x0							
ic_ss_scl_hcnt 0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ic_ss_scl_hcnt RW 0x190															

Register Address Offset	Bit Fields															
ic_ss_scl_lcnt 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ic_ss_scl_lcnt RW 0x1D6															
ic_fs_scl_hcnt 0x1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ic_fs_scl_hcnt RW 0x3C															
ic_fs_scl_lcnt 0x20	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ic_fs_scl_lcnt RW 0x82															
ic_intr_stat 0x2C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		r_master_on_hold RO 0x0	r_restart_det RO 0x0	r_gen_call RO 0x0	r_start_det RO 0x0	r_stop_det RO 0x0	r_activity RO 0x0	r_rx_done RO 0x0	r_tx_abrt RO 0x0	r_rd_req RO 0x0	r_tx_empty RO 0x0	r_tx_over RO 0x0	r_rx_full RO 0x0	r_rx_over RO 0x0	r_rx_under RO 0x0
ic_intr_mask 0x30	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		m_master_on_hold RW 0x0	m_restart_det RW 0x0	m_gen_call RW 0x1	m_start_det RW 0x0	m_stop_det RW 0x0	m_activity RW 0x0	m_rx_done RW 0x1	m_tx_abrt RW 0x1	m_rd_req RW 0x1	m_tx_empty RW 0x1	m_tx_over RW 0x1	m_rx_full RW 0x1	m_rx_over RW 0x1	m_rx_under RW 0x1

Register Address Offset	Bit Fields															
ic_raw_intr_stat 0x34	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		master_on_hold RO 0x0	restart_det RO 0x0	gen_call RO 0x0	start_det RO 0x0	stop_det RO 0x0	activity RO 0x0	rx_done RO 0x0	tx_abrt RO 0x0	rd_req RO 0x0	tx_empty RO 0x0	tx_over RO 0x0	rx_full RO 0x0	rx_over RO 0x0	rx_under RO 0x0
ic_rx_tl 0x38	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								rx_tl RW 0x0							
ic_tx_tl 0x3C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								tx_tl RW 0x0							
ic_clr_intr 0x40	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															clr_intr RO 0x0
ic_clr_rx_under 0x44	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															clr_rx_under RO 0x0
ic_clr_rx_over 0x48	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															clr_rx_over RO 0x0

Register Address Offset	Bit Fields															
ic_clr_tx_over 0x4C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															clr_tx_over RO 0x0
ic_clr_rd_req 0x50	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															clr_rd_req RO 0x0
ic_clr_tx_abrt 0x54	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															clr_tx_abort RO 0x0
ic_clr_rx_done 0x58	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															clr_rx_done RO 0x0
ic_clr_activity 0x5C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															clr_activity RO 0x0
ic_clr_stop_det 0x60	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															clr_stop_det RO 0x0

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
ic_clr_start_det 0x64	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															clr_start_det RO 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
ic_clr_gen_call 0x68	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															clr_gen_call RO 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
ic_enable 0x6C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														txabort RW 0x0	enable RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
ic_status 0x70	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved									slv_activity RO 0x0	mst_activity RO 0x0	rff RO 0x0	rfne RO 0x0	tfe RO 0x1	tfnf RO 0x1	activity RO 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
ic_txflr 0x74	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved									txflr RO 0x0						
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
ic_rxflr 0x78	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved									rxflr RO 0x0						

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ic_sda_hold 0x7C	Reserved								ic_sda_rx_hold RW 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ic_sda_tx_hold RW 0x1															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	tx_flush_cnt RO 0x0									rsvd_ic_tx_abrt_source_22to17 RO 0x0						abrt_user_abrt RO 0x0
ic_tx_abrt_source 0x80	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	abrt_slvrd_intx RO 0x0	abrt_slv_arblost RO 0x0	abrt_slvflush_txfifo RO 0x0	arb_lost RO 0x0	abrt_master_dis RO 0x0	abrt_10b_rd_norstrt RO 0x0	abrt_sbyte_norstrt RO 0x0	abrt_hs_norstrt RO 0x0	abrt_sbyte_ackdet RO 0x0	abrt_hs_ackdet RO 0x0	abrt_gcall_read RO 0x0	abrt_gcall_noack RO 0x0	abrt_txdata_noack RO 0x0	abrt_10addr2_noack RO 0x0	abrt_10addr1_noack RO 0x0	abrt_7b_addr_noack RO 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ic_slv_data_nack_only 0x84	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															nack RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	rsvd_ic_dma_cr_31to2 RO 0x0															
ic_dma_cr 0x88	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_ic_dma_cr_31to2 RO 0x0														tdmae RW 0x0	rdmae RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
ic_dma_tdlr 0x8C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved										dmatdl RW 0x0					

Register Address Offset	Bit Fields															
ic_dma_rdlr 0x90	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved										dmardl RW 0x0					
ic_sda_setup 0x94	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								sda_setup RW 0x64							
ic_ack_general_call 0x98	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	rsvd_ic_ack_gen_31to1 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_ic_ack_gen_31to1 RO 0x0															ack_gen_call RW 0x1
ic_enable_status 0x9C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													slv_rx_data_lost RO 0x0	slv_disabled_while_busy RO 0x0	ic_en RO 0x0
ic_comp_param_1 0xF4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved								tx_buffer_depth RO 0x3F							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rx_buffer_depth RO 0x3F								add_encoded_params RO 0x1	has_dma RO 0x1	intr_io RO 0x1	hc_count_values RO 0x0	max_speed_mode RO 0x2		apb_data_width RO 0x2	

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ic_comp_version 0xF8	ic_comp_version RO 0x3132312A															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ic_comp_version RO 0x3132312A															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ic_comp_type 0xFC	ic_comp_type RO 0x44570140															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ic_comp_type RO 0x44570140															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ic_fs_spklen 0xA0	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								ic_fs_spklen RW 0x2							
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ic_clr_restart_det 0xA8	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															clr_restart_det RO 0x0
i_i2c_emac_2_i2c																
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	rsvd_ic_con_31to10 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ic_con 0x0	rsvd_ic_con_31to10 RO 0x0						rx_fifo_full_hld_ctrl RW 0x0	tx_empty_ctrl RW 0x0	stop_det_ifaddressed RW 0x0	ic_slave_disable RW 0x1	ic_restart_en RW 0x1	ic_10bitaddr_master RO 0x1	ic_10bitaddr_slave RW 0x1	speed RW 0x2		master_mode RW 0x1

Register Address Offset	Bit Fields															
ic_tar 0x4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved			ic_10bitaddr_master RW 0x1	special RW 0x0	gc_or_start RW 0x0	ic_tar RW 0x55									
ic_sar 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						ic_sar RW 0x55									
ic_data_cmd 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved					restart WO 0x0	stop WO 0x0	cmd WO 0x0	dat RW 0x0							
ic_ss_scl_hcnt 0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ic_ss_scl_hcnt RW 0x190															
ic_ss_scl_lcnt 0x18	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ic_ss_scl_lcnt RW 0x1D6															
ic_fs_scl_hcnt 0x1C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ic_fs_scl_hcnt RW 0x3C															

Register Address Offset	Bit Fields															
ic_fs_scl_lcnt 0x20	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ic_fs_scl_lcnt RW 0x82															
ic_intr_stat 0x2C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		r_master_on_hold RO 0x0	r_restart_det RO 0x0	r_gen_call RO 0x0	r_start_det RO 0x0	r_stop_det RO 0x0	r_activity RO 0x0	r_rx_done RO 0x0	r_tx_abrt RO 0x0	r_rd_req RO 0x0	r_tx_empty RO 0x0	r_tx_over RO 0x0	r_rx_full RO 0x0	r_rx_over RO 0x0	r_rx_under RO 0x0
ic_intr_mask 0x30	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		m_master_on_hold RW 0x0	m_restart_det RW 0x0	m_gen_call RW 0x1	m_start_det RW 0x0	m_stop_det RW 0x0	m_activity RW 0x0	m_rx_done RW 0x1	m_tx_abrt RW 0x1	m_rd_req RW 0x1	m_tx_empty RW 0x1	m_tx_over RW 0x1	m_rx_full RW 0x1	m_rx_over RW 0x1	m_rx_under RW 0x1
ic_raw_intr_stat 0x34	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		master_on_hold RO 0x0	restart_det RO 0x0	gen_call RO 0x0	start_det RO 0x0	stop_det RO 0x0	activity RO 0x0	rx_done RO 0x0	tx_abrt RO 0x0	rd_req RO 0x0	tx_empty RO 0x0	tx_over RO 0x0	rx_full RO 0x0	rx_over RO 0x0	rx_under RO 0x0
ic_rx_tl 0x38	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								rx_tl RW 0x0							

Register Address Offset	Bit Fields															
ic_tx_tl 0x3C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								tx_tl RW 0x0							
ic_clr_intr 0x40	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															clr_intr RO 0x0
ic_clr_rx_under 0x44	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															clr_rx_under RO 0x0
ic_clr_rx_over 0x48	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															clr_rx_over RO 0x0
ic_clr_tx_over 0x4C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															clr_tx_over RO 0x0
ic_clr_rd_req 0x50	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															clr_rd_req RO 0x0

Register Address Offset	Bit Fields															
ic_clr_tx_abrt 0x54	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															clr_tx_abort RO 0x0
ic_clr_rx_done 0x58	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															clr_rx_done RO 0x0
ic_clr_activity 0x5C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															clr_activity RO 0x0
ic_clr_stop_det 0x60	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															clr_stop_det RO 0x0
ic_clr_start_det 0x64	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															clr_start_det RO 0x0
ic_clr_gen_call 0x68	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															clr_gen_call RO 0x0

Register Address Offset	Bit Fields															
ic_enable 0x6C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														txabort RW 0x0	enable RW 0x0
ic_status 0x70	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved									slv_activity RO 0x0	mst_activity RO 0x0	rff RO 0x0	rfne RO 0x0	tfe RO 0x1	tfnf RO 0x1	activity RO 0x0
ic_txflr 0x74	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved									txflr RO 0x0						
ic_rxflr 0x78	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved									rxflr RO 0x0						
ic_sda_hold 0x7C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved								ic_sda_rx_hold RW 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ic_sda_tx_hold RW 0x1															

Register Address Offset	Bit Fields															
ic_tx_abrt_source 0x80	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	tx_flush_cnt RO 0x0									rsvd_ic_tx_abrt_source_22to17 RO 0x0						abrt_user_abrt RO 0x0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	abrt_slvrd_intx RO 0x0	abrt_slv_arblost RO 0x0	abrt_slvflush_txfifo RO 0x0	arb_lost RO 0x0	abrt_master_dis RO 0x0	abrt_10b_rd_norstrt RO 0x0	abrt_sbyte_norstrt RO 0x0	abrt_hs_norstrt RO 0x0	abrt_sbyte_ackdet RO 0x0	abrt_hs_ackdet RO 0x0	abrt_gcall_read RO 0x0	abrt_gcall_noack RO 0x0	abrt_txdata_noack RO 0x0	abrt_10addr2_noack RO 0x0	abrt_10addr1_noack RO 0x0	abrt_7b_addr_noack RO 0x0
ic_slv_data_nack_only 0x84	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															nack RW 0x0
ic_dma_cr 0x88	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	rsvd_ic_dma_cr_31to2 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_ic_dma_cr_31to2 RO 0x0														tdmae RW 0x0	rdmae RW 0x0
ic_dma_tdlr 0x8C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved										dmatdl RW 0x0					
ic_dma_rdlr 0x90	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved										dmardl RW 0x0					

Register Address Offset	Bit Fields															
ic_sda_setup 0x94	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								sda_setup RW 0x64							
ic_ack_general_call 0x98	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	rsvd_ic_ack_gen_31to1 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_ic_ack_gen_31to1 RO 0x0															ack_gen_call RW 0x1
ic_enable_status 0x9C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													slv_rx_data_lost RO 0x0	slv_disabled_while_busy RO 0x0	ic_en RO 0x0
ic_comp_param_1 0xF4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved								tx_buffer_depth RO 0x3F							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rx_buffer_depth RO 0x3F								add_encoded_params RO 0x1	has_dma RO 0x1	intr_io RO 0x1	hc_count_values RO 0x0	max_speed_mode RO 0x2		apb_data_width RO 0x2	
ic_comp_version 0xF8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ic_comp_version RO 0x3132312A															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ic_comp_version RO 0x3132312A															

Register Address Offset	Bit Fields															
ic_comp_type 0xFC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ic_comp_type RO 0x44570140															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ic_comp_type RO 0x44570140															
ic_fs_spklen 0xA0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								ic_fs_spklen RW 0x2							
ic_clr_restart_det 0xA8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															clr_restart_det RO 0x0

ic_con

```
I2C Control Register
This register can only be written when the IC_ENABLE[0] register
is set to 0. Writes at other times have no effect.
```

Module Instance	Base Address	Register Address
i_i2c_emac_0_i2c	0xFFC02400	0xFFC02400
i_i2c_emac_1_i2c	0xFFC02500	0xFFC02500
i_i2c_emac_2_i2c	0xFFC02600	0xFFC02600

Offset: 0x0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
rsvd_ic_con_31to10 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rsvd_ic_con_31to9 RO 0x0							tx_empty_ctrl RW 0x0	stop_det_ifaddressed RW 0x0	ic_slave_disable RW 0x1	ic_restart_en RW 0x1	ic_10bitaddr_master RO 0x1	ic_10bitaddr_slave RW 0x1	speed RW 0x2		master_mode RW 0x1

ic_con Fields

Bit	Name	Description	Access	Reset
31:9	rsvd_ic_con_31to9	Reserved bits [31:9] - Read Only	RO	0x0
8	tx_empty_ctrl	This bit controls the generation of the TX_EMPTY interrupt, as described in the IC_RAW_INTR_STAT register. Reset value: 0x0.	RW	0x0
7	stop_det_ifaddressed	In slave mode: 1: issues the STOP_DET interrupt only when it is addressed. 0: issues the STOP_DET irrespective of whether it's addressed or not. Dependencies: This register bit value is applicable in the slave mode only (MASTER_MODE = 1'b0) Reset value: 0x0 NOTE: During a general call address, this slave does not issue the STOP_DET interrupt if STOP_DET_IF_ADDRESSED = 1'b1, even if the slave responds to the general call address by generating ACK. The STOP_DET interrupt is generated only when the transmitted address matches the slave address (SAR).	RW	0x0

Bit	Name	Description	Access	Reset
6	ic_slave_disable	This bit controls whether I2C has its slave disabled, which means once the presetn signal is applied, then this bit is set to 1. When this bit is set (slave is disabled), I2C module functions only as a master and does not perform any action that requires a slave. 0: slave is enabled 1: slave is disabled NOTE: Software should ensure that if this bit is written with 0, then bit 0 should also be written with a 0. **Value** **Description** 0x0 ENABLE 0x1 DISABLE	RW	0x1

Bit	Name	Description	Access	Reset
5	ic_restart_en	Determines whether RESTART conditions may be sent when acting as a master. Some older slaves do not support handling RESTART conditions; however, RESTART conditions are used in several I2C operations. 0: disable 1: enable When RESTART is disabled, the master is prohibited from performing the following functions: - Change direction within a transfer (split) - Send a START BYTE - High-speed mode operation - Combined format transfers in 7-bit addressing modes - Read operation with a 10-bit address - Send multiple bytes per transfer By replacing RESTART condition followed by a STOP and a subsequent START condition, split operations are broken down into multiple I2C transfers. If the above operations are performed, it will result in setting bit 6 (TX_ABRT) of the IC_RAW_INTR_STAT register. Reset value: 0x1 **Value** **Description** 0x0 DISABLE 0x1 ENABLE	RW	0x1
4	ic_10bitaddr_master	This bit is read only. 0: 7-bit addressing 1: 10-bit addressing Reset value: 0x1 **Value** **Description** 0x0 MSTADDR7BIT 0x1 MSTADDR10BIT	RO	0x1

Bit	Name	Description	Access	Reset
3	ic_10bitaddr_slave	When acting as a slave, this bit controls whether the I2C responds to 7- or 10-bit addresses. 0: 7-bit addressing. The I2C ignores transactions that involve 10-bit addressing; for 7-bit addressing, only the lower 7 bits of the IC_SAR register are compared. 1: 10-bit addressing. The I2C responds to only 10-bit addressing transfers that match the full 10 bits of the IC_SAR register. Reset value: 0x1 **Value** / **Description** 0x0 — SLVADDR7BIT 0x1 — SLVADDR10BIT	RW	0x1
2:1	speed	These bits control at which speed the I2C operates; its setting is relevant only if one is operating the I2C in master mode. Hardware protects against illegal values being programmed by software. This register should be programmed only with a value in the range of 1 to 400 kbits/s; otherwise, hardware updates this register with the value of 400 kbits/s. 1: standard mode (100 kbit/s) 2: fast mode (<=400 kbit/s) or fast mode plus (<=1000Kbit/s) Reset value: 0x2 **Value** / **Description** 0x1 — STANDARD 0x2 — FAST	RW	0x2

Bit	Name	Description	Access	Reset
0	master_mode	This bit controls whether the I2C master is enabled. 0: master disabled 1: master enabled Reset value: 0x1 NOTE: Software should ensure that if this bit is written with '1' then bit 6 should also be written with a '1'. **Value** **Description** 0x0 DISABLE 0x1 ENABLE	RW	0x1

ic_tar

```
Name: I2C Target Address Register

All bits can be dynamically updated as long as any set of the
following
conditions are true:
- The I2C is NOT enabled (IC_ENABLE[0] is set to 0);
   or
- The I2C is enabled (IC_ENABLE[0]=1);
   AND
  The I2C is NOT engaged in any Master (tx, rx) operation
(IC_STATUS[5]=0);
   AND
  The I2C is enabled to operate in Master mode (IC_CON[0]=1);
   AND
  there are NO entries in the TX FIFO (IC_STATUS[2]=1)
```

Module Instance	Base Address	Register Address
i_i2c_emac_0_i2c	0xFFC02400	0xFFC02404
i_i2c_emac_1_i2c	0xFFC02500	0xFFC02504
i_i2c_emac_2_i2c	0xFFC02600	0xFFC02604

Offset: 0x4

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			ic_10bit addr_master RW 0x1	special RW 0x0	gc_or_start RW 0x0	ic_tar RW 0x55									

ic_tar Fields

Bit	Name	Description	Access	Reset
12	ic_10bitaddr_master	This bit controls whether the I2C starts its transfers in 7- or 10-bit addressing mode when acting as a master. 0: 7-bit addressing 1: 10-bit addressing Reset value: 0x1 parameter **Value** **Description** 0x0 START7 0x1 START10	RW	0x1
11	special	This bit indicates whether software performs a General Call or START BYTE command. 0: ignore bit 10 GC_OR_START and use IC_TAR normally 1: perform special I2C command as specified in GC_OR_START bit Reset value: 0x0 **Value** **Description** 0x0 GENCALL 0x1 STARTBYTE	RW	0x0

Bit	Name	Description	Access	Reset
10	gc_or_start	If bit 11 (SPECIAL) is set to 1, then this bit indicates whether a General Call or START byte command is to be performed by the I2C. 0: General Call Address after issuing a General Call, only writes may be performed. Attempting to issue a read command results in setting bit 6 (TX_ABRT) of the IC_RAW_INTR_STAT register. The I2C remains in General Call mode until the SPECIAL bit value (bit 11) is cleared. 1: START BYTE Reset value: 0x0 **Value** / **Description** 0x0 GENCALL 0x1 STARTBYTE	RW	0x0
9:0	ic_tar	This is the target address for any master transaction. When transmitting a General Call, these bits are ignored. To generate a START BYTE, the CPU needs to write only once into these bits. Reset value: 0x55 If the IC_TAR and IC_SAR are the same, loopback exists but the FIFOs are shared between master and slave, so full loopback is not feasible. Only one direction loopback mode is supported (simplex), not duplex. A master cannot transmit to itself; it can transmit to only a slave.	RW	0x55

ic_sar

Name: I2C Slave Address Register

Module Instance	Base Address	Register Address
i_i2c_emac_0_i2c	0xFFC02400	0xFFC02408
i_i2c_emac_1_i2c	0xFFC02500	0xFFC02508

Module Instance	Base Address	Register Address
i_i2c_emac_2_i2c	0xFFC02600	0xFFC02608

Offset: 0x8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						ic_sar RW 0x55									

ic_sar Fields

Bit	Name	Description	Access	Reset
9:0	ic_sar	The IC_SAR holds the slave address when the I2C is operating as a slave. For 7-bit addressing, only IC_SAR[6:0] is used. This register can be written only when the I2C interface is disabled, which corresponds to the IC_ENABLE[0] register being set to 0. Writes at other times have no effect. Note: The default values cannot be any of the reserved address locations: that is, 0x00 to 0x07, or 0x78 to 0x7f. The correct operation of the device is not guaranteed if you program the IC_SAR or IC_TAR to a reserved value. Reset value: 0x55	RW	0x55

ic_data_cmd

I2C Rx/Tx Data Buffer and Command Register;

```
      this is the register the CPU writes to when
      filling the TX FIFO and the CPU reads from when
      retrieving bytes from RX FIFO
Size:
11 bits (writes), 8 bits (read)
```

Module Instance	Base Address	Register Address
i_i2c_emac_0_i2c	0xFFC02400	0xFFC02410
i_i2c_emac_1_i2c	0xFFC02500	0xFFC02510
i_i2c_emac_2_i2c	0xFFC02600	0xFFC02610

Offset: 0x10

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					restart WO 0x0	stop WO 0x0	cmd WO 0x0	dat RW 0x0							

ic_data_cmd Fields

Bit	Name	Description	Access	Reset
10	restart	This bit controls whether a RESTART is issued before the byte is sent or received. 1 - If IC_RESTART_EN is 1, a RESTART is issued before the data is sent/received (according to the value of CMD), regardless of whether or not the transfer direction is changing from the previous command; if IC_RESTART_EN is 0, a STOP followed by a START is issued instead. 0 - If IC_RESTART_EN is 1, a RESTART is issued only if the transfer direction is changing from the previous command; if IC_RESTART_EN is 0, a STOP followed by a START is issued instead. Reset value: 0x0	WO	0x0
9	stop	This bit controls whether a STOP is issued after the byte is sent or received. 1 - STOP is issued after this byte, regardless of whether or not the Tx FIFO is empty. If the Tx FIFO is not empty, the master immediately tries to start a new transfer by issuing a START and arbitrating for the bus. 0 - STOP is not issued after this byte, regardless of whether or not the Tx FIFO is empty. If the Tx FIFO is not empty, the master continues the current transfer by sending/receiving data bytes according to the value of the CMD bit. If the Tx FIFO is empty, the master holds the SCL line low and stalls the bus until a new command is available in the Tx FIFO. Reset value: 0x0	WO	0x0

Bit	Name	Description	Access	Reset
8	cmd	This bit controls whether a read or a write is performed. This bit does not control the direction when the DW_apb_i2c acts as a slave. It controls only the direction when it acts as a master. 1 = Read 0 = Write When a command is entered in the TX FIFO, this bit distinguishes the write and read commands. In slave-receiver mode, this bit is a 'don't care' because writes to this register are not required. In slave-transmitter mode, a '0' indicates that CPU data is to be transmitted and as DAT or IC_DATA_CMD[7:0]. When programming this bit, you should remember the following: attempting to perform a read operation after a General Call command has been sent results in a TX_ABRT interrupt (bit 6 of the IC_RAW_INTR_STAT register), unless bit 11 (SPECIAL) in the IC_TAR register has been cleared. If a '1' is written to this bit after receiving a RD_REQ interrupt, then a TX_ABRT interrupt occurs. NOTE: It is possible that while attempting a master I2C read transfer on the I2C, a RD_REQ interrupt may have occurred simultaneously due to a remote I2C master addressing the I2C module. In this type of scenario,the I2C module ignores the IC_DATA_CMD write, generates a TX_ABRT interrupt, and waits to service the RD_REQ interrupt. Reset value: 0x0 **Value** **Description** 0x0 WR	WO	0x0

Bit	Name	Description	Access	Reset
		Value Description 0x1 RD		
7:0	dat	This register contains the data to be transmitted or received on the I2C bus. If you are writing to this register and want to perform a read, bits 7:0 (DAT) are ignored by the I2C moudle. However, when you read this register, these bits return the value of data received on the the I2C interface. Reset value: 0x0	RW	0x0

ic_ss_scl_hcnt

Standard Speed I2C Clock SCL High Count Register

Module Instance	Base Address	Register Address
i_i2c_emac_0_i2c	0xFFC02400	0xFFC02414
i_i2c_emac_1_i2c	0xFFC02500	0xFFC02514
i_i2c_emac_2_i2c	0xFFC02600	0xFFC02614

Offset: 0x14

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ic_ss_scl_hcnt RW 0x190															

ic_ss_scl_hcnt Fields

Bit	Name	Description	Access	Reset
15:0	ic_ss_scl_hcnt	This register must be set before any I2C bus transaction can take place to ensure proper I/O timing. This register sets the SCL clock high-period count for standard speed. This register can be written only when the I2C interface is disabled which corresponds to the IC_ENABLE[0] register being set to 0. Writes at other times have no effect. The minimum valid value is 6; hardware prevents values less than this being written, and if attempted results in 6 being set. NOTE: This register must not be programmed to a value higher than 65525, because the I2C module uses a 16-bit counter to flag an I2C bus idle condition when this counter reaches a value of 0x019A.	RW	0x190

ic_ss_scl_lcnt

Name: Standard Speed I2C Clock SCL Low Count Register

Module Instance	Base Address	Register Address
i_i2c_emac_0_i2c	0xFFC02400	0xFFC02418
i_i2c_emac_1_i2c	0xFFC02500	0xFFC02518
i_i2c_emac_2_i2c	0xFFC02600	0xFFC02618

Offset: 0x18

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ic_ss_scl_lcnt RW 0x1D6															

ic_ss_scl_lcnt Fields

Bit	Name	Description	Access	Reset
15:0	ic_ss_scl_lcnt	This register must be set before any I2C bus transaction can take place to ensure proper I/O timing. This register sets the SCL clock low period count for standard speed. This register can be written only when the I2C interface is disabled which corresponds to the IC_ENABLE[0] register being set to 0. Writes at other times have no effect. The minimum valid value is 8; hardware prevents values less than this being written, and if attempted, results in 8 being set. Reset value: 0x1D6	RW	0x1D6

ic_fs_scl_hcnt

Fast Mode or Fast Mode Plus I2C Clock SCL High Count Register

Module Instance	Base Address	Register Address
i_i2c_emac_0_i2c	0xFFC02400	0xFFC0241C
i_i2c_emac_1_i2c	0xFFC02500	0xFFC0251C
i_i2c_emac_2_i2c	0xFFC02600	0xFFC0261C

Offset: 0x1C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ic_fs_scl_hcnt RW 0x3C															

ic_fs_scl_hcnt Fields

Bit	Name	Description	Access	Reset
15:0	ic_fs_scl_hcnt	This register must be set before any I2C bus transaction can take place to ensure proper I/O timing. This register sets the SCL clock high-period count for fast mode or fast mode plus. It is used in high-speed mode to send the Master Code and START BYTE or General CALL. This register can be written only when the I2C interface is disabled, which corresponds to the IC_ENABLE[0] register being set to 0. Writes at other times have no effect. The minimum valid value is 6; hardware prevents values less than this being written, and if attempted results in 6 being set. Reset value: 0x003C	RW	0x3C

ic_fs_scl_lcnt

Fast Mode or Fast Mode Plus I2C Clock SCL Low Count Register

Module Instance	Base Address	Register Address
i_i2c_emac_0_i2c	0xFFC02400	0xFFC02420
i_i2c_emac_1_i2c	0xFFC02500	0xFFC02520

Module Instance	Base Address	Register Address
i_i2c_emac_2_i2c	0xFFC02600	0xFFC02620

Offset: 0x20

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ic_fs_scl_lcnt RW 0x82															

ic_fs_scl_lcnt Fields

Bit	Name	Description	Access	Reset
15:0	ic_fs_scl_lcnt	This register must be set before any I2C bus transaction can take place to ensure proper I/O timing. This register sets the SCL clock low period count for fast speed. This register can be written only when the I2C interface is disabled, which corresponds to the IC_ENABLE[0] register being set to 0. Writes at other times have no effect. The minimum valid value is 8; hardware prevents values less than this being written, and if attempted results in 8 being set. Reset value: 0x0082	RW	0x82

ic_intr_stat

I2C Interrupt Status Register

Each bit in this register has a corresponding mask bit in the IC_INTR_MASK register. These bits are cleared by reading the matching interrupt clear register. The unmasked raw versions of these bits are available in the IC_RAW_INTR_STAT register.

Module Instance	Base Address	Register Address
i_i2c_emac_0_i2c	0xFFC02400	0xFFC0242C
i_i2c_emac_1_i2c	0xFFC02500	0xFFC0252C
i_i2c_emac_2_i2c	0xFFC02600	0xFFC0262C

Offset: 0x2C

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		r_master_on_hold RO 0x0	r_restart_det RO 0x0	r_gen_call RO 0x0	r_start_det RO 0x0	r_stop_det RO 0x0	r_activity RO 0x0	r_rx_done RO 0x0	r_tx_abrt RO 0x0	r_rd_req RO 0x0	r_tx_empty RO 0x0	r_tx_over RO 0x0	r_rx_full RO 0x0	r_rx_over RO 0x0	r_rx_under RO 0x0

ic_intr_stat Fields

Bit	Name	Description	Access	Reset
13	r_master_on_hold	Indicates whether master is holding the bus and TX FIFO is empty. Reset value: 0x0	RO	0x0
12	r_restart_det	Indicates a RESTART condition has occurred on the I2C interface when the I2C module is operating in slave mode and addressed. Reset value: 0x0	RO	0x0

Bit	Name	Description	Access	Reset
11	r_gen_call	Set only when a General Call address is received and it is acknowledged. It stays set until it is cleared either by disabling the I2C or when the CPU reads bit 0 of the IC_CLR_GEN_CALL register. The I2C stores the received data in the Rx buffer. Reset value: 0x0	RO	0x0
10	r_start_det	Indicates whether a START or RESTART condition has occurred on the I2C interface regardless of whether the I2C module is operating in slave or master mode. Reset value: 0x0	RO	0x0

Bit	Name	Description	Access	Reset
9	r_stop_det	The behavior of the STOP_DET interrupt status differs based on the STOP_DET_IFADDRESSED selection in the IC_CON register When STOP_DET_IFADDRESSED =0 : Indicates whether a STOP condition has occurred on the I2C interface regardless of whether the I2C module is operating in slave or master mode. In slave mode, a STOP_DET interrupt is generated irrespective of whether the slave is addressed or not. When STOP_DET_IFADDRESSED = 1 : In Master Mode (MASTER_MODE = 1'b1) , indicates a STOP condition has occured on the I2C interface. In Slave Mode (MASTER_MODE = 1'b0) ,STOP_DET interrupt is generated only if the slave is addressed. NOTE: During a general call address, this slave does not issue a STOP_DET interrupt if STOP_DET_ IFADDRESSED=1'b1, even if the slave responds to the general call address by generating ACK. The STOP_DET interrupt is generated only when the transmitted address matches the slave address (SAR). Reset value: 0x0	RO	0x0

Bit	Name	Description	Access	Reset
8	r_activity	This bit captures the I2C module activity and stays set until it is cleared. There are four ways to clear it: - Disabling the I2C module - Reading the IC_CLR_ACTIVITY register - Reading the IC_CLR_INTR register - System reset Once this bit is set, it stays set unless one of the four methods is used to clear it. Even if the I2C module is idle, this bit remains set until cleared, indicating that there was activity on the bus. Reset value: 0x0	RO	0x0
7	r_rx_done	When the I2C module is acting as a slave-transmitter, this bit is set to 1 if the master does not acknowledge a transmitted byte. This occurs on the last byte of the transmission, indicating that the transmission is done. Reset value: 0x0	RO	0x0
6	r_tx_abrt	This bit indicates if the I2C module, as an I2C transmitter, is unable to complete the intended actions on the contents of the transmit FIFO. This situation can occur both as an I2C master or an I2C slave, and is referred to as a 'transmit abort'. When this bit is set to 1, the IC_TX_ABRT_SOURCE register indicates the reason why the transmit abort takes places. NOTE: The I2C module flushes/resets/empties the TX FIFO whenever this bit is set. The TX FIFO remains in this flushed state until the register IC_CLR_TX_ABRT is read. Once this read is performed, the TX FIFO is then ready to accept more data bytes from the APB interface. Reset value: 0x0	RO	0x0

Bit	Name	Description	Access	Reset
5	r_rd_req	This bit is set to 1 when the I2C module is acting as a slave and another I2C master is attempting to read data from the I2C module. The HPS I2C holds the I2C bus in a wait state (SCL=0) until this interrupt is serviced, which means that the slave has been addressed by a remote master that is asking for data to be transferred. The processor must respond to this interrupt and then write the requested data to the IC_DATA_CMD register. This bit is set to 0 just after the processor reads the IC_CLR_RD_REQ register. Reset value: 0x0	RO	0x0

Bit	Name	Description	Access	Reset
4	r_tx_empty	The behavior of the TX_EMPTY interrupt status differs based on the TX_EMPTY_CTRL selection in the IC_CON register. When TX_EMPTY_CTRL = 0: This bit is set to 1 when the transmit buffer is at or below the threshold value set in the IC_TX_TL register. When TX_EMPTY_CTRL = 1: This bit is set to 1 when the transmit buffer is at or below the threshold value set in the IC_TX_TL register and the transmission of the address/data from the internal shift register for the most recently popped command is completed. It is automatically cleared by hardware when the buffer level goes above the threshold. When IC_ENABLE[0] is set to 0, the TX FIFO is flushed and held in reset. There the TX FIFO looks like it has no data within it, so this bit is set to 1, provided there is activity in the master or slave state machines. When there is no longer any activity, then with ic_en=0, this bit is set to 0. Reset value: 0x0	RO	0x0
3	r_tx_over	Set during transmit if the transmit buffer is filled to the maximum buffer depth (64) and the processor attempts to issue another I2C command by writing to the IC_DATA_CMD register. When the module is disabled, this bit keeps its level until the master or slave state machines go into idle, and when ic_en goes to 0, this interrupt is cleared. Reset value: 0x0	RO	0x0

Bit	Name	Description	Access	Reset
2	r_rx_full	Set when the receive buffer reaches or goes above the RX_TL threshold in the IC_RX_TL register. It is automatically cleared by hardware when buffer level goes below the threshold. If the module is disabled (IC_ENABLE[0]=0), the RX FIFO is flushed and held in reset; therefore the RX FIFO is not full. So this bit is cleared once the IC_ENABLE bit 0 is programmed with a 0, regardless of the activity that continues. Reset value: 0x0	RO	0x0
1	r_rx_over	Set if the receive buffer is completely filled to its maximum receive buffer depth (64) and an additional byte is received from an external I2C device. The I2C module acknowledges this, but any data bytes received after the FIFO is full are lost. If the module is disabled (IC_ENABLE[0]=0), this bit keeps its level until the master or slave state machines go into idle, and when ic_en goes to 0, this interrupt is cleared. Reset value: 0x0	RO	0x0
0	r_rx_under	Set if the processor attempts to read the receive buffer when it is empty by reading from the IC_DATA_CMD register. If the module is disabled (IC_ENABLE[0]=0), this bit keeps its level until the master or slave state machines go into idle, and when ic_en goes to 0, this interrupt is cleared. Reset value: 0x0	RO	0x0

ic_intr_mask

I2C Interrupt Mask Register

These bits mask their corresponding interrupt status bits. They are active high; a value of 0 prevents a bit from generating an interrupt.

Module Instance	Base Address	Register Address
i_i2c_emac_0_i2c	0xFFC02400	0xFFC02430
i_i2c_emac_1_i2c	0xFFC02500	0xFFC02530
i_i2c_emac_2_i2c	0xFFC02600	0xFFC02630

Offset: 0x30

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		m_master_on_hold RW 0x0	m_restart_det RW 0x0	m_gen_call RW 0x1	m_start_det RW 0x0	m_stop_det RW 0x0	m_activity RW 0x0	m_rx_done RW 0x1	m_tx_abrt RW 0x1	m_rd_req RW 0x1	m_tx_empty RW 0x1	m_tx_over RW 0x1	m_rx_full RW 0x1	m_rx_over RW 0x1	m_rx_under RW 0x1

ic_intr_mask Fields

Bit	Name	Description	Access	Reset
13	m_master_on_hold	This bit masks the R_MASTER_ON_HOLD interrupt in IC_INTR_STAT register. Reset value: 0x0	RW	0x0
12	m_restart_det	This bit masks the R_RESTART_DET interrupt in IC_INTR_STAT register. Reset value: 0x0	RW	0x0

Bit	Name	Description	Access	Reset
11	m_gen_call	This bit masks the R_GEN_CALL interrupt in IC_INTR_STAT register. Reset value: 0x1	RW	0x1
10	m_start_det	This bit masks the R_START_DET interrupt in IC_INTR_STAT register. Reset value: 0x0	RW	0x0
9	m_stop_det	This bit masks the R_STOP_DET interrupt in IC_INTR_STAT register. Reset value: 0x0	RW	0x0
8	m_activity	This bit masks the R_ACTIVITY interrupt in IC_INTR_STAT register. Reset value: 0x0	RW	0x0
7	m_rx_done	This bit masks the R_RX_DONE interrupt in IC_INTR_STAT register. Reset value: 0x1	RW	0x1
6	m_tx_abrt	This bit masks the R_TX_ABRT interrupt in IC_INTR_STAT register. Reset value: 0x1	RW	0x1
5	m_rd_req	This bit masks the R_RD_REQ interrupt in IC_INTR_STAT register. Reset value: 0x1	RW	0x1
4	m_tx_empty	This bit masks the R_TX_EMPTY interrupt in IC_INTR_STAT register. Reset value: 0x1	RW	0x1
3	m_tx_over	This bit masks the R_TX_OVER interrupt in IC_INTR_STAT register. Reset value: 0x1	RW	0x1
2	m_rx_full	This bit masks the R_RX_FULL interrupt in IC_INTR_STAT register. Reset value: 0x1	RW	0x1
1	m_rx_over	This bit masks the R_RX_OVER interrupt in IC_INTR_STAT register. Reset value: 0x1	RW	0x1
0	m_rx_under	This bit masks the R_RX_UNDER interrupt in IC_INTR_STAT register. Reset value: 0x1	RW	0x1

ic_raw_intr_stat

I2C Raw Interrupt Status Register

Unlike the IC_INTR_STAT register, these bits are not masked so they always show the true status of the I2C Module.

Module Instance	Base Address	Register Address
i_i2c_emac_0_i2c	0xFFC02400	0xFFC02434
i_i2c_emac_1_i2c	0xFFC02500	0xFFC02534
i_i2c_emac_2_i2c	0xFFC02600	0xFFC02634

Offset: 0x34

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		master_on_hold RO 0x0	restart_det RO 0x0	gen_call RO 0x0	start_det RO 0x0	stop_det RO 0x0	activity RO 0x0	rx_done RO 0x0	tx_abrt RO 0x0	rd_req RO 0x0	tx_empty RO 0x0	tx_over RO 0x0	rx_full RO 0x0	rx_over RO 0x0	rx_under RO 0x0

ic_raw_intr_stat Fields

Bit	Name	Description	Access	Reset
13	master_on_hold	Indicates whether master is holding the bus and TX FIFO is empty. Reset value: 0x0	RO	0x0

Bit	Name	Description	Access	Reset
12	restart_det	Indicates whether a RESTART condition has occurred on the I2C interface when the I2C module is operating in Slave mode and the slave is being addressed. (Note:Following are exceptions where the Restart interrupt will not get generated. In the case of Startbyte transfer, where the Restart comes before the Address field as per the I2C protocol defined format, the Slave is still not in the addressed mode and hence will not generate the RESTART_DET interrupt.) Reset value: 0x0	RO	0x0
11	gen_call	Set only when a General Call address is received and it is acknowledged. It stays set until it is cleared either by disabling the I2C module or when the CPU reads bit 0 of the IC_CLR_GEN_CALL register. The I2C stores the received data in the Rx buffer. Reset value: 0x0	RO	0x0
10	start_det	Indicates whether a START or RESTART condition has occurred on the I2C interface regardless of whether the I2C is operating in slave or master mode. Reset value: 0x0	RO	0x0

Bit	Name	Description	Access	Reset
9	stop_det	The behavior of the STOP_DET interrupt status differs based on the STOP_DET_IFADDRESSED selection in the IC_CON register When STOP_DET_IFADDRESSED =0 Indicates whether a STOP condition has occurred on the I2C interface regardless of whether DW_apb_i2c is operating in slave or master mode. In slave mode, a STOP_DET interrupt is generated irrespective of whether the slave is addressed or not. When STOP_DET_IFADDRESSED = 1 In Master Mode (MASTER_MODE = 1), indicates a STOP condition has occured on the I2C interface. In Slave Mode (MASTER_MODE = 0), a STOP_DET interrupt is generated only if the slave is addressed. NOTE: During a general call address, this slave does not issue a STOP_DET interrupt if STOP_DET_IFADDRESSED=1, even if the slave responds to the general call address by generating ACK. The STOP_DET interrupt is generated only when the transmitted address matches the slave address (SAR). Reset value: 0x0.	RO	0x0

Bit	Name	Description	Access	Reset
8	activity	This bit captures the I2C module activity and stays set until it is cleared. There are four ways to clear it: - Disabling the I2C module - Reading the IC_CLR_ACTIVITY register - Reading the IC_CLR_INTR register - System reset Once this bit is set, it stays set unless one of the four methods is used to clear it. Even if the I2C module is idle, this bit remains set until cleared, indicating that there was activity on the bus. Reset value: 0x0	RO	0x0
7	rx_done	When the I2C module is acting as a slave-transmitter, this bit is set to 1 if the master does not acknowledge a transmitted byte. This occurs on the last byte of the transmission, indicating that the transmission is done. Reset value: 0x0	RO	0x0
6	tx_abrt	This bit indicates if the I2C module, as an I2C transmitter, is unable to complete the intended actions on the contents of the transmit FIFO. This situation can occur both as an I2C master or an I2C slave, and is referred to as a 'transmit abort'. When this bit is set to 1, the IC_TX_ABRT_SOURCE register indicates the reason why the transmit abort takes places. NOTE: The I2C module flushes/resets/empties the TX FIFO whenever this bit is set. The TX FIFO remains in this flushed state until the register IC_CLR_TX_ABRT is read. Once this read is performed, the TX FIFO is then ready to accept more data bytes from the APB interface. Reset value: 0x0	RO	0x0

Bit	Name	Description	Access	Reset
5	rd_req	This bit is set to 1 when the I2C module is acting as a slave and another I2C master is attempting to read data from the I2C module. The I2C module holds the I2C bus in a wait state (SCL=0) until this interrupt is serviced, which means that the slave has been addressed by a remote master that is asking for data to be transferred. The processor must respond to this interrupt and then write the requested data to the IC_DATA_CMD register. This bit is set to 0 just after the processor reads the IC_CLR_RD_REQ register. Reset value: 0x0	RO	0x0

Bit	Name	Description	Access	Reset
4	tx_empty	The behavior of the TX_EMPTY interrupt status differs based on the TX_EMPTY_CTRL selection in the IC_CON register. When TX_EMPTY_CTRL = 0: This bit is set to 1 when the transmit buffer is at or below the threshold value set in the IC_TX_TL register. When TX_EMPTY_CTRL = 1: This bit is set to 1 when the transmit buffer is at or below the threshold value. set in the IC_TX_TL register and the transmission of the address/ data from the internal shift register for the most recently popped command is completed. It is automatically cleared by hardware when the buffer level goes above the threshold. When IC_ENABLE[0] is set to 0, the TX FIFO is flushed and held in reset. There the TX FIFO looks like it has no data within it, so this bit is set to 1, provided there is activity in the master or slave state machines. When there is no longer any activity, then with ic_ en=0, this bit is set to 0. Reset value: 0x0.	RO	0x0
3	tx_over	Set during transmit if the transmit buffer is filled to the maximum TxFIFO buffer depth (64) and the processor attempts to issue another I2C command by writing to the IC_DATA_CMD register. When the module is disabled, this bit keeps its level until the master or slave state machines go into idle, and when ic_ en goes to 0, this interrupt is cleared. Reset value: 0x0	RO	0x0

Bit	Name	Description	Access	Reset
2	rx_full	Set when the receive buffer reaches or goes above the RX_TL threshold in the IC_RX_TL register. It is automatically cleared by hardware when buffer level goes below the threshold. If the module is disabled (IC_ENABLE[0]=0), the RX FIFO is flushed and held in reset; therefore the RX FIFO is not full. So this bit is cleared once the IC_ENABLE bit 0 is programmed with a 0, regardless of the activity that continues. Reset value: 0x0	RO	0x0
1	rx_over	Set if the receive buffer is completely filled to the maximum Rx FIFO buffer depth (64) and an additional byte is received from an external I2C device. The I2C module acknowledges this, but any data bytes received after the FIFO is full are lost. If the module is disabled (IC_ENABLE[0]=0), this bit keeps its level until the master or slave state machines go into idle, and when ic_en goes to 0, this interrupt is cleared. Reset value: 0x0	RO	0x0
0	rx_under	Set if the processor attempts to read the receive buffer when it is empty by reading from the IC_DATA_CMD register. If the module is disabled (IC_ENABLE[0]=0), this bit keeps its level until the master or slave state machines go into idle, and when ic_en goes to 0, this interrupt is cleared. Reset value: 0x0	RO	0x0

ic_rx_tl

I2C Receive FIFO Threshold Register

Module Instance	Base Address	Register Address
i_i2c_emac_0_i2c	0xFFC02400	0xFFC02438
i_i2c_emac_1_i2c	0xFFC02500	0xFFC02538
i_i2c_emac_2_i2c	0xFFC02600	0xFFC02638

Offset: 0x38

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								rx_tl RW 0x0							

ic_rx_tl Fields

Bit	Name	Description	Access	Reset
7:0	rx_tl	Receive FIFO Threshold Level Controls the level of entries (or above) that triggers the RX_FULL interrupt (bit 2 in IC_RAW_INTR_STAT register). The valid range is 0-255, with the additional restriction that hardware does not allow this value to be set to a value larger than the depth of the buffer. If an attempt is made to do that, the actual value set will be the maximum depth of the buffer. A value of 0 sets the threshold for 1 entry, and a value of 255 sets the threshold for 256 entries.	RW	0x0

ic_tx_tl

I2C Transmit FIFO Threshold Register

Module Instance	Base Address	Register Address
i_i2c_emac_0_i2c	0xFFC02400	0xFFC0243C
i_i2c_emac_1_i2c	0xFFC02500	0xFFC0253C
i_i2c_emac_2_i2c	0xFFC02600	0xFFC0263C

Offset: 0x3C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								tx_tl RW 0x0							

ic_tx_tl Fields

Bit	Name	Description	Access	Reset
7:0	tx_tl	Transmit FIFO Threshold Level Controls the level of entries (or below) that trigger the TX_EMPTY interrupt (bit 4 in IC_RAW_INTR_STAT register). The valid range is 0-255, with the additional restriction that it may not be set to value larger than the depth of the buffer. If an attempt is made to do that, the actual value set will be the maximum depth of the buffer (64). A value of 0 sets the threshold for 0 entries, and a value of 255 sets the threshold for 255 entries.	RW	0x0

ic_clr_intr

Clear Combined and Individual Interrupt Register

Module Instance	Base Address	Register Address
i_i2c_emac_0_i2c	0xFFC02400	0xFFC02440
i_i2c_emac_1_i2c	0xFFC02500	0xFFC02540
i_i2c_emac_2_i2c	0xFFC02600	0xFFC02640

Offset: 0x40

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															clr_intr RO 0x0

ic_clr_intr Fields

Bit	Name	Description	Access	Reset
0	clr_intr	Read this register to clear the combined interrupt, all individual interrupts, and the IC_TX_ABRT_SOURCE register. This bit does not clear hardware clearable interrupts but software clearable interrupts. Refer to Bit 9 of the IC_TX_ABRT_SOURCE register for an exception to clearing IC_TX_ABRT_SOURCE. Reset value: 0x0	RO	0x0

ic_clr_rx_under

Clear RX_UNDER Interrupt Register

Module Instance	Base Address	Register Address
i_i2c_emac_0_i2c	0xFFC02400	0xFFC02444
i_i2c_emac_1_i2c	0xFFC02500	0xFFC02544
i_i2c_emac_2_i2c	0xFFC02600	0xFFC02644

Offset: 0x44

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															clr_rx_under RO 0x0

ic_clr_rx_under Fields

Bit	Name	Description	Access	Reset
0	clr_rx_under	Read this register to clear the RX_UNDER interrupt (bit 0) of the IC_RAW_INTR_STAT register. Reset value: 0x0	RO	0x0

ic_clr_rx_over

Clear RX_OVER Interrupt Register

Module Instance	Base Address	Register Address
i_i2c_emac_0_i2c	0xFFC02400	0xFFC02448
i_i2c_emac_1_i2c	0xFFC02500	0xFFC02548
i_i2c_emac_2_i2c	0xFFC02600	0xFFC02648

Offset: 0x48

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															clr_rx_over RO 0x0

ic_clr_rx_over Fields

Bit	Name	Description	Access	Reset
0	clr_rx_over	Read this register to clear the RX_OVER interrupt (bit 1) of the IC_RAW_INTR_STAT register. Reset value: 0x0	RO	0x0

ic_clr_tx_over

Clear TX_OVER Interrupt Register

Module Instance	Base Address	Register Address
i_i2c_emac_0_i2c	0xFFC02400	0xFFC0244C
i_i2c_emac_1_i2c	0xFFC02500	0xFFC0254C
i_i2c_emac_2_i2c	0xFFC02600	0xFFC0264C

Offset: 0x4C

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															clr_tx_over RO 0x0

ic_clr_tx_over Fields

Bit	Name	Description	Access	Reset
0	clr_tx_over	Read this register to clear the TX_OVER interrupt (bit 3) of the IC_RAW_INTR_STAT register. Reset value: 0x0	RO	0x0

ic_clr_rd_req

Clear RD_REQ Interrupt Register

Module Instance	Base Address	Register Address
i_i2c_emac_0_i2c	0xFFC02400	0xFFC02450
i_i2c_emac_1_i2c	0xFFC02500	0xFFC02550
i_i2c_emac_2_i2c	0xFFC02600	0xFFC02650

Offset: 0x50

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															clr_rd_req RO 0x0

ic_clr_rd_req Fields

Bit	Name	Description	Access	Reset
0	clr_rd_req	Read this register to clear the RD_REQ interrupt (bit 5) of the IC_RAW_INTR_STAT register. Reset value: 0x0	RO	0x0

ic_clr_tx_abrt

Clear TX_ABRT Interrupt Register

Module Instance	Base Address	Register Address
i_i2c_emac_0_i2c	0xFFC02400	0xFFC02454
i_i2c_emac_1_i2c	0xFFC02500	0xFFC02554
i_i2c_emac_2_i2c	0xFFC02600	0xFFC02654

Offset: 0x54

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															clr_tx_abort RO 0x0

ic_clr_tx_abrt Fields

Bit	Name	Description	Access	Reset
0	clr_tx_abort	Read this register to clear the TX_ABRT interrupt (bit 6) of the IC_RAW_INTR_STAT register, and the IC_TX_ABRT_SOURCE register. This also releases the TX FIFO from the flushed/reset state, allowing more writes to the TX FIFO. Refer to Bit 9 of the IC_TX_ABRT_SOURCE register for an exception to clearing IC_TX_ABRT_SOURCE. Reset value: 0x0	RO	0x0

ic_clr_rx_done

Clear RX_DONE Interrupt Register

Module Instance	Base Address	Register Address
i_i2c_emac_0_i2c	0xFFC02400	0xFFC02458
i_i2c_emac_1_i2c	0xFFC02500	0xFFC02558
i_i2c_emac_2_i2c	0xFFC02600	0xFFC02658

Offset: 0x58

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															clr_rx_done RO 0x0

ic_clr_rx_done Fields

Bit	Name	Description	Access	Reset
0	clr_rx_done	Read this register to clear the RX_DONE interrupt (bit 7) of the IC_RAW_INTR_STAT register. Reset value: 0x0	RO	0x0

ic_clr_activity

Clear ACTIVITY Interrupt Register

Module Instance	Base Address	Register Address
i_i2c_emac_0_i2c	0xFFC02400	0xFFC0245C
i_i2c_emac_1_i2c	0xFFC02500	0xFFC0255C
i_i2c_emac_2_i2c	0xFFC02600	0xFFC0265C

Offset: 0x5C

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															clr_activity RO 0x0

ic_clr_activity Fields

Bit	Name	Description	Access	Reset
0	clr_activity	Reading this register clears the ACTIVITY interrupt if the I2C is not active anymore. If the I2C module is still active on the bus, the ACTIVITY interrupt bit continues to be set. It is automatically cleared by hardware if the module is disabled and if there is no further activity on the bus. The value read from this register to get status of the ACTIVITY interrupt (bit 8) of the IC_RAW_INTR_STAT register. Reset value: 0x0	RO	0x0

ic_clr_stop_det

Clear STOP_DET Interrupt Register

Module Instance	Base Address	Register Address
i_i2c_emac_0_i2c	0xFFC02400	0xFFC02460
i_i2c_emac_1_i2c	0xFFC02500	0xFFC02560
i_i2c_emac_2_i2c	0xFFC02600	0xFFC02660

Offset: 0x60

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															clr_stop_det RO 0x0

ic_clr_stop_det Fields

Bit	Name	Description	Access	Reset
0	clr_stop_det	Read this register to clear the STOP_DET interrupt (bit 9) of the IC_RAW_INTR_STAT register. Reset value: 0x0	RO	0x0

ic_clr_start_det

Clear START_DET Interrupt Register

Module Instance	Base Address	Register Address
i_i2c_emac_0_i2c	0xFFC02400	0xFFC02464
i_i2c_emac_1_i2c	0xFFC02500	0xFFC02564
i_i2c_emac_2_i2c	0xFFC02600	0xFFC02664

Offset: 0x64

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															clr_start_det RO 0x0

ic_clr_start_det Fields

Bit	Name	Description	Access	Reset
0	clr_start_det	Read this register to clear the START_DET interrupt (bit 10) of the IC_RAW_INTR_STAT register. Reset value: 0x0	RO	0x0

ic_clr_gen_call

Name: Clear GEN_CALL Interrupt Register

Module Instance	Base Address	Register Address
i_i2c_emac_0_i2c	0xFFC02400	0xFFC02468
i_i2c_emac_1_i2c	0xFFC02500	0xFFC02568
i_i2c_emac_2_i2c	0xFFC02600	0xFFC02668

Offset: 0x68

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															clr_gen_call RO 0x0

ic_clr_gen_call Fields

Bit	Name	Description	Access	Reset
0	clr_gen_call	Read this register to clear the GEN_CALL interrupt (bit 11) of IC_RAW_INTR_STAT register. Reset value: 0x0	RO	0x0

ic_enable

I2C Enable Register

Module Instance	Base Address	Register Address
i_i2c_emac_0_i2c	0xFFC02400	0xFFC0246C
i_i2c_emac_1_i2c	0xFFC02500	0xFFC0256C
i_i2c_emac_2_i2c	0xFFC02600	0xFFC0266C

Offset: 0x6C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														txabort RW 0x0	enable RW 0x0

ic_enable Fields

Bit	Name	Description	Access	Reset
1	txabort	When set, the controller initiates the transfer abort. 0: ABORT not initiated or ABORT done 1: ABORT operation in progress The software can abort the I2C transfer in master mode by setting this bit. The software can set this bit only when ENABLE is already set; otherwise, the controller ignores any write to ABORT bit. The software cannot clear the ABORT bit once set. In response to an ABORT, the controller issues a STOP and flushes the Tx FIFO after completing the current transfer, then sets the TX_ABORT interrupt after the abort operation. The ABORT bit is cleared automatically after the abort operation. Reset value: 0x0	RW	0x0

Bit	Name	Description	Access	Reset
0	enable	Controls whether the I2C module is enabled. 0: Disables the I2C module (TX and RX FIFOs are held in an erased state) 1: Enables the I2C module Software can disable the I2C module while it is active. However, it is important that care be taken to ensure that the I2C is disabled properly. When I2C module is disabled, the following occurs: - The TX FIFO and RX FIFO get flushed. - Status bits in the IC_INTR_STAT register are still active until I2C goes into IDLE state. If the module is transmitting, it stops as well as deletes the contents of the transmit buffer after the current transfer is complete. If the module is receiving, the I2C stops the current transfer at the end of the current byte and does not acknowledge the transfer. Reset value: 0x0 **Value** **Description** 0x0 DISABLE 0x1 ENABLE	RW	0x0

ic_status

```
          I2C Status Register

This is a read-only register used to indicate the current
transfer status and FIFO status. The status register may be
read at any time. None of the bits in this register request
an interrupt.
When the I2C is disabled by writing 0 in bit 0 of the
IC_ENABLE register:
- Bits 1 and 2 are set to 1
- Bits 3 and 4 are set to 0
When the master or slave state machines goes to idle
and ic_en=0:
```

- Bits 5 and 6 are set to 0

Module Instance	Base Address	Register Address
i_i2c_emac_0_i2c	0xFFC02400	0xFFC02470
i_i2c_emac_1_i2c	0xFFC02500	0xFFC02570
i_i2c_emac_2_i2c	0xFFC02600	0xFFC02670

Offset: 0x70

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									slv_activity RO 0x0	mst_activity RO 0x0	rff RO 0x0	rfne RO 0x0	tfe RO 0x1	tfnf RO 0x1	activity RO 0x0

ic_status Fields

Bit	Name	Description	Access	Reset
6	slv_activity	Slave FSM Activity Status. When the Slave Finite State Machine (FSM) is not in the IDLE state, this bit is set. 0: Slave FSM is in IDLE state so the Slave part of I2C module is not Active 1: Slave FSM is not in IDLE state so the Slave part of the I2C module is Active Reset value: 0x0 **Value** **Description** 0x0 IDLE 0x1 NOTIDLE	RO	0x0

Bit	Name	Description	Access	Reset
5	mst_activity	Master FSM Activity Status. When the Master Finite State Machine (FSM) is not in the IDLE state, this bit is set. 0: Master FSM is in IDLE state so the Master part of the I2C module is not Active 1: Master FSM is not in IDLE state so the Master part of the I2C module is Active Note IC_STATUS[0]-that is, ACTIVITY bit-is the OR of SLV_ACTIVITY and MST_ACTIVITY bits. Reset value: 0x0 **Value** **Description** 0x0 IDLE 0x1 NOTIDLE	RO	0x0
4	rff	Receive FIFO Completely Full. When the receive FIFO is completely full, this bit is set. When the receive FIFO contains one or more empty location, this bit is cleared. 0: Receive FIFO is not full 1: Receive FIFO is full Reset value: 0x0 **Value** **Description** 0x0 NOTFULL 0x1 FULL	RO	0x0

Bit	Name	Description	Access	Reset
3	rfne	Receive FIFO Not Empty. This bit is set when the receive FIFO contains one or more entries; it is cleared when the receive FIFO is empty. 0: Receive FIFO is empty 1: Receive FIFO is not empty Reset value: 0x0 **Value** / **Description** 0x0 — EMPTY 0x1 — NOTEMPTY	RO	0x0
2	tfe	Transmit FIFO Completely Empty. When the transmit FIFO is completely empty, this bit is set. When it contains one or more valid entries, this bit is cleared. This bit field does not request an interrupt. 0: Transmit FIFO is not empty 1: Transmit FIFO is empty Reset value: 0x1 **Value** / **Description** 0x0 — NOTEMPTY 0x1 — EMPTY	RO	0x1
1	tfnf	Transmit FIFO Not Full. Set when the transmit FIFO contains one or more empty locations, and is cleared when the FIFO is full. 0: Transmit FIFO is full 1: Transmit FIFO is not full Reset value: 0x1 **Value** / **Description** 0x0 — FULL 0x1 — NOTFULL	RO	0x1
0	activity	I2C Activity Status. Reset value: 0x0	RO	0x0

ic_txflr

```
Name: I2C Transmit FIFO Level Register

This register contains the number of valid data
entries in the transmit FIFO buffer. It is cleared
whenever:
- The I2C is disabled
- There is a transmit abort that is, TX_ABRT bit is
  set in the IC_RAW_INTR_STAT register
- The slave bulk transmit mode is aborted
The register increments whenever data is placed into
the transmit FIFO and decrements when data is
taken from the transmit FIFO.
```

Module Instance	Base Address	Register Address
i_i2c_emac_0_i2c	0xFFC02400	0xFFC02474
i_i2c_emac_1_i2c	0xFFC02500	0xFFC02574
i_i2c_emac_2_i2c	0xFFC02600	0xFFC02674

Offset: 0x74

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									txflr RO 0x0						

ic_txflr Fields

Bit	Name	Description	Access	Reset
6:0	txflr	Transmit FIFO Level. Contains the number of valid data entries in the transmit FIFO. Reset value: 0x0	RO	0x0

ic_rxflr

```
I2C Receive FIFO Level Register

This register contains the number of valid data
entries in the receive FIFO buffer. It is cleared
whenever:
- The I2C is disabled
- Whenever there is a transmit abort caused by any
  of the events tracked in IC_TX_ABRT_SOURCE
The register increments whenever data is placed into
the receive FIFO and decrements when data is
taken from the receive FIFO.
```

Module Instance	Base Address	Register Address
i_i2c_emac_0_i2c	0xFFC02400	0xFFC02478
i_i2c_emac_1_i2c	0xFFC02500	0xFFC02578
i_i2c_emac_2_i2c	0xFFC02600	0xFFC02678

Offset: 0x78

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									rxflr RO 0x0						

ic_rxflr Fields

Bit	Name	Description	Access	Reset
6:0	rxflr	Receive FIFO Level. Contains the number of valid data entries in the receive FIFO. Reset value: 0x0	RO	0x0

ic_sda_hold

```
I2C SDA Hold Time Length Register

The bits [15:0] of this register are used to control the hold time of
SDA during
transmit in both slave and master mode (after SCL goes from HIGH to
LOW).
The bits [23:16] of this register are used to extend the SDA
transition (if any)
whenever SCL is HIGH in the receiver in either master or slave mode.
The values in this register are in units of l4_sp_clk period.
This register controls the amount of time delay.
The relevant I2C requirement is thd:DAT as detailed in the I2C
Bus Specification.
```

Module Instance	Base Address	Register Address
i_i2c_emac_0_i2c	0xFFC02400	0xFFC0247C
i_i2c_emac_1_i2c	0xFFC02500	0xFFC0257C
i_i2c_emac_2_i2c	0xFFC02600	0xFFC0267C

Offset: 0x7C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								ic_sda_rx_hold RW 0x0							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ic_sda_tx_hold RW 0x1															

ic_sda_hold Fields

Bit	Name	Description	Access	Reset
23:16	ic_sda_rx_hold	Sets the required SDA hold time in units of l4_sp_clk period, when the I2C module acts as a receiver.	RW	0x0

Bit	Name	Description	Access	Reset
15:0	ic_sda_tx_hold	Sets the required SDA hold time in units of l4_sp_clk period, when the I2C module acts as a transmitter.	RW	0x1

ic_tx_abrt_source

I2C Transmit Abort Source Register

This register has 32 bits that indicate the source of the TX_ABRT bit. Except for Bit 9, this register is cleared whenever the IC_CLR_TX_ABRT register or the IC_CLR_INTR register is read. To clear Bit 9, the source of the ABRT_SBYTE_NORSTRT must be fixed first; RESTART must be enabled (IC_CON[5]=1), the SPECIAL bit must be cleared (IC_TAR[11]), or the GC_OR_START bit must be cleared (IC_TAR[10]). Once the source of the ABRT_SBYTE_NORSTRT is fixed, then this bit can be cleared in the same manner as other bits in this register. If the source of the ABRT_SBYTE_NORSTRT is not fixed before attempting to clear this bit, Bit 9 clears for one cycle and is then re-asserted.

Module Instance	Base Address	Register Address
i_i2c_emac_0_i2c	0xFFC02400	0xFFC02480
i_i2c_emac_1_i2c	0xFFC02500	0xFFC02580
i_i2c_emac_2_i2c	0xFFC02600	0xFFC02680

Offset: 0x80

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
tx_flush_cnt RO 0x0									rsvd_ic_tx_abrt_source_22to17 RO 0x0						abrt_user_abrt RO 0x0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
abrt_slvrd_intx RO 0x0	abrt_slv_arblost RO 0x0	abrt_slvflush_txfifo RO 0x0	arb_lost RO 0x0	abrt_master_dis RO 0x0	abrt_10b_rd_norstrt RO 0x0	abrt_sbyte_norstrt RO 0x0	abrt_hs_norstrt RO 0x0	abrt_sbyte_ackdet RO 0x0	abrt_hs_ackdet RO 0x0	abrt_gcall_read RO 0x0	abrt_gcall_noack RO 0x0	abrt_txdata_noack RO 0x0	abrt_10addr2_noack RO 0x0	abrt_10addr1_noack RO 0x0	abrt_7b_addr_noack RO 0x0

ic_tx_abrt_source Fields

Bit	Name	Description	Access	Reset
31:23	tx_flush_cnt	This field indicates the number of Tx FIFO Data Commands which are flushed due to TX_ABRT interrupt. It is cleared whenever I2C is disabled. Reset value: 0x0 Role of the I2C module: Master-Transmitter or Slave-Transmitter	RO	0x0
22:17	rsvd_ic_tx_abrt_source_22to17	Reserved Reset value: 0x0	RO	0x0
16	abrt_user_abrt	This is a master-mode-only bit. Master has detected the transfer abort (IC_ENABLE[1]) Reset value: 0x0 Role of the I2C module: Master-Transmitter	RO	0x0
15	abrt_slvrd_intx	1: When the processor side responds to a slave mode request for data to be transmitted to a remote master and user writes a 1 in CMD (bit 8) of IC_DATA_CMD register. Reset value: 0x0 Role of the I2C module: Slave-Transmitter	RO	0x0

Bit	Name	Description	Access	Reset
14	abrt_slv_arblost	1: Slave lost the bus while transmitting data to a remote master. IC_TX_ABRT_SOURCE[12] is set at the same time. Note: Even though the slave never 'owns' the bus, something could go wrong on the bus. This is a fail safe check. For instance, during a data transmission at the low-to-high transition of SCL, if what is on the data bus is not what is supposed to be transmitted, then the I2C module no longer owns the bus. Reset value: 0x0 Role of the I2C module: Slave-Transmitter	RO	0x0
13	abrt_slvflush_txfifo	1: Slave has received a read command and some data exists in the TX FIFO so the slave issues a TX_ABRT interrupt to flush old data in TX FIFO. Reset value: 0x0 Role of the I2C module: Slave-Transmitter	RO	0x0
12	arb_lost	1: Master has lost arbitration, or if IC_TX_ABRT_SOURCE[14] is also set, then the slave transmitter has lost arbitration. Reset value: 0x0 Role of the I2C module: Master-Transmitter or Slave-Transmitter	RO	0x0
11	abrt_master_dis	1: User tries to initiate a Master operation with the Master mode disabled. Reset value: 0x0 Role of the I2C module: Master-Transmitter or Master-Receiver	RO	0x0

Bit	Name	Description	Access	Reset
10	abrt_10b_rd_norstrt	1: The restart is disabled (IC_RESTART_EN bit (IC_CON[5]) =0) and the master sends a read command in 10-bit addressing mode. Reset value: 0x0 Role of the I2C module: Master-Receiver	RO	0x0
9	abrt_sbyte_norstrt	To clear Bit 9, the source of the ABRT_SBYTE_NORSTRT must be fixed first; restart must be enabled (IC_CON[5]=1), the SPECIAL bit must be cleared (IC_TAR[11]), or the GC_OR_START bit must be cleared (IC_TAR[10]). Once the source of the ABRT_SBYTE_NORSTRT is fixed, then this bit can be cleared in the same manner as other bits in this register. If the source of the ABRT_SBYTE_NORSTRT is not fixed before attempting to clear this bit, bit 9 clears for one cycle and then gets reasserted. 1: The restart is disabled (IC_RESTART_EN bit (IC_CON[5]) =0) and the user is trying to send a START Byte. Reset value: 0x0 Role of the I2C module: Master	RO	0x0
8	abrt_hs_norstrt	1: The restart is disabled (IC_RESTART_EN bit (IC_CON[5]) =0) and the user is trying to use the master to transfer data in High Speed mode. Reset value: 0x0 Role of the I2C module: Master-Transmitter or Master-Receiver	RO	0x0

Bit	Name	Description	Access	Reset
7	abrt_sbyte_ackdet	1: Master has sent a START Byte and the START Byte was acknowledged (wrong behavior). Reset value: 0x0 Role of the I2C module: Master	RO	0x0
6	abrt_hs_ackdet	1: Master is in High Speed mode and the High Speed Master code was acknowledged (wrong behavior). Reset value: 0x0 Role of the I2C module: Master	RO	0x0
5	abrt_gcall_read	1: The I2C module in master mode sent a General Call but the user programmed the byte following the General Call to be a read from the bus (IC_DATA_CMD[9] is set to 1). Reset value: 0x0 Role of the I2C module: Master-Transmitter	RO	0x0
4	abrt_gcall_noack	1: The I2C module in master mode sent a General Call and no slave on the bus acknowledged the General Call. Reset value: 0x0 Role of the I2C module: Master-Transmitter	RO	0x0
3	abrt_txdata_noack	1: This is a master-mode only bit. Master has received an acknowledgement for the address, but when it sent data byte(s) following the address, it did not receive an acknowledge from the remote slave(s). Reset value: 0x0 Role of the I2C module: Master-Transmitter	RO	0x0

Bit	Name	Description	Access	Reset
2	abrt_10addr2_noack	1: Master is in 10-bit address mode and the second address byte of the 10-bit address was not acknowledged by any slave. Reset value: 0x0 Role of the I2C module: Master-Transmitter or Master-Receiver	RO	0x0
1	abrt_10addr1_noack	1: Master is in 10-bit address mode and the first 10-bit address byte was not acknowledged by any slave. Reset value: 0x0 Role of the I2C module: Master-Transmitter or Master-Receiver	RO	0x0
0	abrt_7b_addr_noack	1: Master is in 7-bit addressing mode and the address sent was not acknowledged by any slave. Reset value: 0x0 Role of the I2C module: Master-Transmitter or Master-Receiver	RO	0x0

ic_slv_data_nack_only

Generate Slave Data NACK Register

The register is used to generate a NACK for the data part of a transfer when the I2C module is acting as a slave-receiver.

Module Instance	Base Address	Register Address
i_i2c_emac_0_i2c	0xFFC02400	0xFFC02484
i_i2c_emac_1_i2c	0xFFC02500	0xFFC02584
i_i2c_emac_2_i2c	0xFFC02600	0xFFC02684

Offset: 0x84

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															nack RW 0x0

ic_slv_data_nack_only Fields

Bit	Name	Description	Access	Reset
0	nack	Generate NACK. This NACK generation only occurs when the I2C module is a slave-receiver. If this register is set to a value of 1, it can only generate a NACK after a data byte is received; hence, the data transfer is aborted and the data received is not pushed to the receive buffer. When the register is set to a value of 0, it generates NACK/ACK, depending on normal criteria. 1 = generate NACK after data byte received 0 = generate NACK/ACK normally Reset value: 0x0 **Value** **Description** 0x0 NORM 0x1 AFTERDBYTE	RW	0x0

ic_dma_cr

DMA Control Register

The register is used to enable the DMA Controller interface operation. There is a separate bit for transmit and receive. This can be

programmed
regardless of the state of IC_ENABLE.

Module Instance	Base Address	Register Address
i_i2c_emac_0_i2c	0xFFC02400	0xFFC02488
i_i2c_emac_1_i2c	0xFFC02500	0xFFC02588
i_i2c_emac_2_i2c	0xFFC02600	0xFFC02688

Offset: 0x88

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
rsvd_ic_dma_cr_31to2 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rsvd_ic_dma_cr_31to2 RO 0x0														tdmae RW 0x0	rdmae RW 0x0

ic_dma_cr Fields

Bit	Name	Description	Access	Reset
31:2	rsvd_ic_dma_cr_31to2	Reserved bits [31:1] - Read Only	RO	0x0
1	tdmae	Transmit DMA Enable. //This bit enables/disables the transmit FIFO DMA channel. 0 = Transmit DMA disabled 1 = Transmit DMA enabled Reset value: 0x0 **Value** **Description** 0x0 DISABLE 0x1 ENABLE	RW	0x0

Bit	Name	Description	Access	Reset
0	rdmae	Receive DMA Enable. This bit enables/disables the receive FIFO DMA channel. 0 = Receive DMA disabled 1 = Receive DMA enabled Reset value: 0x0 Value: Description 0x0: DISABLE 0x1: ENABLE	RW	0x0

ic_dma_tdlr

Name: DMA Transmit Data Level Register

Module Instance	Base Address	Register Address
i_i2c_emac_0_i2c	0xFFC02400	0xFFC0248C
i_i2c_emac_1_i2c	0xFFC02500	0xFFC0258C
i_i2c_emac_2_i2c	0xFFC02600	0xFFC0268C

Offset: 0x8C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										dmatdl RW 0x0					

ic_dma_tdlr Fields

Bit	Name	Description	Access	Reset
5:0	dmatdl	Transmit Data Level. This bit field controls the level at which a DMA request is made by the transmit logic. It is equal to the watermark level; that is, the dma_tx_req signal is generated when the number of valid data entries in the transmit FIFO is equal to or below this field value, and TDMAE = 1. Reset value: 0x0	RW	0x0

ic_dma_rdlr

I2C Receive Data Level Register

Module Instance	Base Address	Register Address
i_i2c_emac_0_i2c	0xFFC02400	0xFFC02490
i_i2c_emac_1_i2c	0xFFC02500	0xFFC02590
i_i2c_emac_2_i2c	0xFFC02600	0xFFC02690

Offset: 0x90

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										dmardl RW 0x0					

ic_dma_rdlr Fields

Bit	Name	Description	Access	Reset
5:0	dmardl	Receive Data Level. This bit field controls the level at which a DMA request is made by the receive logic. The watermark level = DMARDL+1; that is, dma_rx_req is generated when the number of valid data entries in the receive FIFO is equal to or more than this field value + 1, and RDMAE =1. For instance, when DMARDL is 0, then dma_rx_req is asserted when 1 or more data entries are present in the receive FIFO. Reset value: 0x0	RW	0x0

ic_sda_setup

I2C SDA Setup Register

This register controls the amount of time delay
(in terms of number of l4_sp_clk clock periods) introduced
in the rising edge of SCL, relative to SDA changing, when
the I2C module services a read request in a slave-transmitter
operation.
The relevant I2C requirement is tSU:DAT (note 4) as detailed in the
I2C Bus Specification.

Module Instance	Base Address	Register Address
i_i2c_emac_0_i2c	0xFFC02400	0xFFC02494
i_i2c_emac_1_i2c	0xFFC02500	0xFFC02594
i_i2c_emac_2_i2c	0xFFC02600	0xFFC02694

Offset: 0x94

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								sda_setup RW 0x64							

ic_sda_setup Fields

Bit	Name	Description	Access	Reset
7:0	sda_setup	SDA Setup. It is recommended that if the required delay is 1000ns, then for an l4_sp_clk frequency of 10 MHz, IC_SDA_SETUP should be programmed to a value of 11. Default Reset value: 0x64	RW	0x64

ic_ack_general_call

I2C ACK General Call Register

The register controls whether the I2C responds with a ACK or NACK when it receives an I2C General Call address.
Note :This register is applicable only when the I2C module is in slave mode

Module Instance	Base Address	Register Address
i_i2c_emac_0_i2c	0xFFC02400	0xFFC02498
i_i2c_emac_1_i2c	0xFFC02500	0xFFC02598
i_i2c_emac_2_i2c	0xFFC02600	0xFFC02698

Offset: 0x98

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
rsvd_ic_ack_gen_31to1 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rsvd_ic_ack_gen_31to1 RO 0x0															ack_gen_call RW 0x1

ic_ack_general_call Fields

Bit	Name	Description	Access	Reset
31:1	rsvd_ic_ack_gen_31to1	Reserved bits [31:1] - Read Only	RO	0x0
0	ack_gen_call	ACK General Call. When set to 1, the I2C module responds with a ACK (by asserting ic_data_oe) when it receives a General Call. Otherwise, the I2C module responds with a NACK (by negating ic_data_oe). Default Reset value: 0x1 **Value** **Description** 0x0 NACK 0x1 ACK	RW	0x1

ic_enable_status

I2C Enable Status Register

The register is used to report the I2C hardware status when the IC_ENABLE[0] register is set from 1 to 0; that is, when the I2C module is disabled.
If IC_ENABLE[0] has been set to 1, bits 2:1 are forced to 0, and bit 0 is forced to 1.
If IC_ENABLE[0] has been set to 0, bits 2:1 is only be valid as soon as bit 0 is read as '0'.
Note
When IC_ENABLE[0] has been written with '0'a delay occurs for bit 0 to be read as '0' because disabling the I2C module depends on I2C bus activities.

Module Instance	Base Address	Register Address
i_i2c_emac_0_i2c	0xFFC02400	0xFFC0249C
i_i2c_emac_1_i2c	0xFFC02500	0xFFC0259C
i_i2c_emac_2_i2c	0xFFC02600	0xFFC0269C

Offset: 0x9C

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													slv_rx_data_lost RO 0x0	slv_disabled_while_busy RO 0x0	ic_en RO 0x0

ic_enable_status Fields

Bit	Name	Description	Access	Reset
2	slv_rx_data_lost	Slave Received Data Lost. This bit indicates if a Slave-Receiver operation has been aborted with at least one data byte received from an I2C transfer due to the setting bit 0 of IC_ENABLE from 1 to 0. When read as 1, I2C module is deemed to have been actively engaged in an aborted I2C transfer (with matching address) and the data phase of the I2C transfer has been entered, even though a data byte has been responded with a NACK. NOTE: If the remote I2C master terminates the transfer with a STOP condition before the I2C module has a chance to NACK a transfer, and IC_ENABLE[0] has been set to 0, then this bit is also set to 1. When read as 0, the I2C is deemed to have been disabled without being actively involved in the data phase of a Slave-Receiver transfer. NOTE: The CPU can safely read this bit when IC_EN (bit 0) is read as 0. Reset value: 0x0	RO	0x0

Bit	Name	Description	Access	Reset
1	slv_disabled_while_busy	Slave Disabled While Busy (Transmit, Receive). This bit indicates if a potential or active Slave operation has been aborted due to the setting bit 0 of the IC_ENABLE register from 1 to 0. This bit is set when the CPU writes a 0 to the IC_ENABLE register while: (a) the I2C is receiving the address byte of the Slave-Transmitter operation from a remote master; OR, (b) address and data bytes of the Slave-Receiver operation from a remote master. When read as 1, the I2C module is deemed to have forced a NACK during any part of an I2C transfer, irrespective of whether the I2C address matches the slave address set in the I2C (IC_SAR register) OR if the transfer is completed before IC_ENABLE is set to 0 but has not taken effect. NOTE: If the remote I2C master terminates the transfer with a STOP condition before the I2C module has a chance to NACK a transfer, and IC_ENABLE[0] has been set to 0, then this bit will also be set to 1. When read as 0, the I2C mdoule is deemed to have been disabled when there is master activity, or when the I2C bus is idle. NOTE: The CPU can safely read this bit when IC_EN (bit 0) is read as 0. Reset value: 0x0	RO	0x0

Bit	Name	Description	Access	Reset
0	ic_en	ic_en Status. This bit always reflects the value driven on the output port ic_en. When read as 1, the I2C module is deemed to be in an enabled state. When read as 0, the I2C module is deemed completely inactive. NOTE: The CPU can safely read this bit anytime. When this bit is read as 0, the CPU can safely read SLV_RX_DATA_LOST (bit 2) and SLV_DISABLED_WHILE_BUSY (bit 1). Reset value: 0x0	RO	0x0

ic_comp_param_1

Name: Component Parameter Register 1

Note
This is a constant read-only register that contains encoded information about the component's parameter settings.

Module Instance	Base Address	Register Address
i_i2c_emac_0_i2c	0xFFC02400	0xFFC024F4
i_i2c_emac_1_i2c	0xFFC02500	0xFFC025F4
i_i2c_emac_2_i2c	0xFFC02600	0xFFC026F4

Offset: 0xF4

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								tx_buffer_depth RO 0x3F							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rx_buffer_depth RO 0x3F								add_encoded_params RO 0x1	has_dma RO 0x1	intr_io RO 0x1	hc_count_values RO 0x0	max_speed_mode RO 0x2		apb_data_width RO 0x2	

ic_comp_param_1 Fields

Bit	Name	Description	Access	Reset
23:16	tx_buffer_depth	This field identifies the Tx Buffer depth **Value** **Description** 0x40 FIFO64BYTES	RO	0x3F
15:8	rx_buffer_depth	This field identifies the Rx Buffer Depth **Value** **Description** 0x40 FIFO64BYTES	RO	0x3F
7	add_encoded_params	Reading 1 in this bit means that the capability of reading these encoded parameters via software has been included. Otherwise, the entire register is 0 regardless of the setting of any other parameters that are encoded in the bits. 0: False 1: True **Value** **Description** 0x1 ADDENCPARAMS	RO	0x1

Bit	Name	Description	Access	Reset
6	has_dma	This field identifies if the DMA is present. 0: False 1: True Value: 0x1 — Description: PRESENT	RO	0x1
5	intr_io	This bit indicates if interrupt outputs are individual or combined into a single output. Value: 0x1 — Description: COMBINED	RO	0x1
4	hc_count_values	This field indicates if the CNT registers are read-only or read/write. Value: 0x0 — Description: READWRITE	RO	0x0
3:2	max_speed_mode	This field indicates the maximum I2C bus speed supported. Value: 0x2 — Description: FAST	RO	0x2
1:0	apb_data_width	This field identifies the width of the APB bus. Value: 0x2 — Description: WIDTH32BITS	RO	0x2

ic_comp_version

I2C Component Version Register

Module Instance	Base Address	Register Address
i_i2c_emac_0_i2c	0xFFC02400	0xFFC024F8
i_i2c_emac_1_i2c	0xFFC02500	0xFFC025F8

Module Instance	Base Address	Register Address
i_i2c_emac_2_i2c	0xFFC02600	0xFFC026F8

Offset: 0xF8

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ic_comp_version RO 0x3132312A															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ic_comp_version RO 0x3132312A															

ic_comp_version Fields

Bit	Name	Description	Access	Reset
31:0	ic_comp_version	**Value** **Description** 0x3132312a VER_1_21A	RO	0x3132312A

ic_comp_type

I2C Component Type Register

Module Instance	Base Address	Register Address
i_i2c_emac_0_i2c	0xFFC02400	0xFFC024FC
i_i2c_emac_1_i2c	0xFFC02500	0xFFC025FC
i_i2c_emac_2_i2c	0xFFC02600	0xFFC026FC

Offset: 0xFC

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ic_comp_type RO 0x44570140															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ic_comp_type RO 0x44570140															

ic_comp_type Fields

Bit	Name	Description	Access	Reset
31:0	ic_comp_type	Designware Component Type number = 0x44_57_01_40. This assigned unique hex value is constant and is derived from the two ASCII letters 'DW' followed by a 16-bit unsigned number.	RO	0x44570140

ic_fs_spklen

I2C SS, FS or FM+ spike suppression limit

This register is used to store the duration, measured in l4_sp_clk cycles, of the longest spike that is filtered out by the spike suppression logic when the component is operating in SS, FS or FM+ modes. The relevant I2C requirement is tSP (table 4) as detailed in the I2C Bus Specification. This register must be programmed with a minimum value of 1.

Module Instance	Base Address	Register Address
i_i2c_emac_0_i2c	0xFFC02400	0xFFC024A0
i_i2c_emac_1_i2c	0xFFC02500	0xFFC025A0
i_i2c_emac_2_i2c	0xFFC02600	0xFFC026A0

Offset: 0xA0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								ic_fs_spklen RW 0x2							

ic_fs_spklen Fields

Bit	Name	Description	Access	Reset
7:0	ic_fs_spklen	This register must be set before any I2C bus transaction can take place to ensure stable operation. This register sets the duration, measured in l4_sp_clk cycles, of the longest spike in the SCL or SDA lines that will be filtered out by the spike suppression logic. This register can be written only when the I2C interface is disabled which corresponds to the IC_ENABLE[0] register being set to 0. Writes at other times have no effect. The minimum valid value is 1; hardware prevents values less than this being written, and if attempted results in 1 being set.	RW	0x2

ic_clr_restart_det

Name: Clear RESTART_DET Interrupt Register

Module Instance	Base Address	Register Address
i_i2c_emac_0_i2c	0xFFC02400	0xFFC024A8
i_i2c_emac_1_i2c	0xFFC02500	0xFFC025A8
i_i2c_emac_2_i2c	0xFFC02600	0xFFC026A8

Offset: 0xA8

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															clr_restart_det RO 0x0

ic_clr_restart_det Fields

Bit	Name	Description	Access	Reset
0	clr_restart_det	Read this register to clear the RESTART_DET interrupt (bit 12) of IC_RAW_INTR_STAT register. Reset value: 0x0	RO	0x0

Document Revision History

Table 20-5: Document Revision History

Date	Version	Changes
October 2016	2016.10.28	Maintenance release.
May 2016	2016.05.27	Maintenance release.
May 2016	2016.05.03	Maintenance release.
November 2015	2015.11.02	• Renamed *Interface Pins* section to *I²C Controller Signal Description* and moved section below *I²C Controller Block Diagram and System Integration*
May 2015	2015.05.04	• Added *Impact of SCL Rise Time and Fall Time On Generated SCL* figure to Clock Synchronization section • Updated Minimum High and Low Counts section
December 2014	2014.12.15	• Maintenance release. • Added *Taking the I²C Out of Reset* section.

Date	Version	Changes
August 2014	2014.08.18	Initial release.

UART Controller 21

2016.10.28

a10_5v4 Subscribe Send Feedback

The hard processor system (HPS) provides two UART controllers for asynchronous serial communication. The UART controllers are based on an industry standard 16550 UART controller. The UART controllers are instances of the Synopsys® DesignWare® APB Universal Asynchronous Receiver/Transmitter (DW_apb_uart) peripheral. (54)

UART Controller Features

The UART controller provides the following functionality and features:

- Programmable character properties, such as number of data bits per character, optional parity bits, and number of stop bits †
- Line break generation and detection †
- DMA controller handshaking interface
- Prioritized interrupt identification †
- Programmable baud rate
- False start bit detection †
- Automatic flow control mode per 16750 standard †
- Internal loopback mode support
- 128-byte transmit and receive FIFO buffers
  - FIFO buffer status registers †
  - FIFO buffer access mode (for FIFO buffer testing) enables write of receive FIFO buffer by master and read of transmit FIFO buffer by master †
- Shadow registers reduce software overhead and provide programmable reset †
- Transmitter holding register empty (THRE) interrupt mode †
- Separate thresholds for DMA request and handshake signals to maximize throughput

(54) Portions © 2016 Synopsys, Inc. Used with permission. All rights reserved. Synopsys & DesignWare are registered trademarks of Synopsys, Inc. All documentation is provided "as is" and without any warranty. Synopsys expressly disclaims any and all warranties, express, implied, or otherwise, including the implied warranties of merchantability, fitness for a particular purpose, and non-infringement, and any warranties arising out of a course of dealing or usage of trade.

†Paragraphs marked with the dagger (†) symbol are Synopsys Proprietary. Used with permission.

UART Controller Block Diagram and System Integration

Figure 21-1: UART Block Diagram

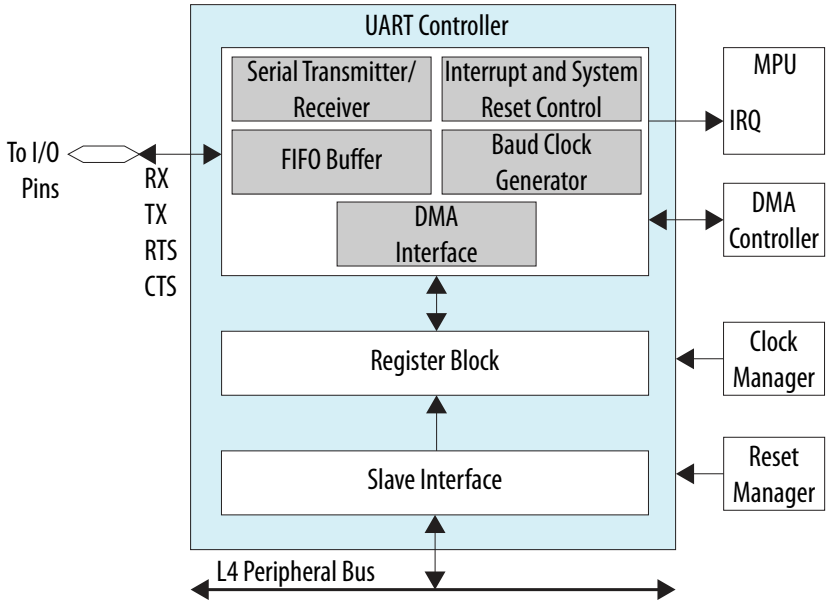


Table 21-1: UART Controller Block Descriptions

Block	Description
Slave interface	Slave interface between the component and L4 peripheral bus.
Register block	Provides main UART control, status, and interrupt generation functions.†
FIFO buffer	Provides FIFO buffer control and storage. †
Baud clock generator	Generates the transmitter and receiver baud clock. With a reference clock of 100 MHz, the UART controller supports transfer rates of 95 baud to 6.25 Mbaud. This supports communication with all known 16550 devices. The baud rate is controlled by programming the interrupt enable or divisor latch high (`IER_DLH`) and receive buffer, transmit holding, or divisor latch low (`RBR_THR_DLL`) registers.
Serial transmitter	Converts parallel data written to the UART into serial data and adds all additional bits, as specified by the control register, for transmission. This makeup of serial data, referred to as a character, exits the block in serial UART. †

Block	Description
Serial receiver	Converts the serial data character (as specified by the control register) received in the UART format to parallel form. Parity error detection, framing error detection and line break detection is carried out in this block. †
DMA interface	The UART controller includes a DMA controller interface to indicate when received data is available or when the transmit FIFO buffer requires data. The DMA requires two channels, one for transmit and one for receive. The UART controller supports single and burst transfers. You can use DMA in FIFO buffer and non-FIFO buffer mode.

Related Information

DMA Controller on page 16-1
For more information, refer to the DMA Controller chapter.

UART Controller Signal Description

HPS I/O Pins

There are two UARTs available in the HPS. Signals from both sets of UARTs can be routed to the HPS I/O. For more information on routing UART signals to the HPS I/O, refer to the *HPS Component Interfaces* chapter.

Table 21-2: HPS I/O UART Pin Descriptions

Pin	Width	Direction	Description
RX	1 bit	Input	Serial Input
TX	1 bit	Output	Serial Output
CTS	1 bit	Input	Clear to send
RTS	1 bit	Output	Request to send

Related Information

HPS Component Interfaces on page 28-14
For more information on how to route the UART signals to the FPGA and HPS I/O, refer to this chapter.

FPGA Routing

There are two UARTs provided in the HPS. Both sets of UART signals can be routed to the FPGA. For more information on routing UART signals to the FPGA, refer to the *HPS Component Interfaces* chapter.

Table 21-3: Signals for FPGA Routing

Signal	Width	Direction	Description
uart_rxd	1 bit	Input	Serial input
uart_txd	1 bit	Output	Serial output
uart_cts	1 bit	Input	Clear to send
uart_rts	1 bit	Output	Request to send
uart_dsr	1 bit	Input	Data set ready
uart_dcd	1 bit	Input	Data carrier detect
uart_ri	1 bit	Input	Ring indicator
uart_dtr	1 bit	Output	Data terminal ready
uart_out1_n	1 bit	Output	User defined output 1
uart_out2_n	1 bit	Output	User defined output 2

Related Information

HPS Component Interfaces on page 28-14
For more information on how to route the UART signals to the FPGA and HPS I/O, refer to this chapter.

Functional Description of the UART Controller

The HPS UART is based on an industry-standard 16550 UART. The UART supports serial communication with a peripheral, modem (data carrier equipment), or data set. The master (CPU) writes data over the slave bus to the UART. The UART converts the data to serial format and transmits to the destination device. The UART also receives serial data and stores it for the master (CPU). †

The UART's registers control the character length, baud rate, parity generation and checking, and interrupt generation. The UART's single interrupt output signal is supported by several prioritized interrupt types that trigger assertion. You can separately enable or disable each of the interrupt types with the control registers. †

FIFO Buffer Support

The UART controller includes 128-byte FIFO buffers to buffer transmit and receive data. FIFO buffer access mode allows the master to write the receive FIFO buffer and to read the transmit FIFO buffer for test purposes. FIFO buffer access mode is enabled with the FIFO access register (FAR). Once enabled, the

control portions of the transmit and receive FIFO buffers are reset and the FIFO buffers are treated as empty. †

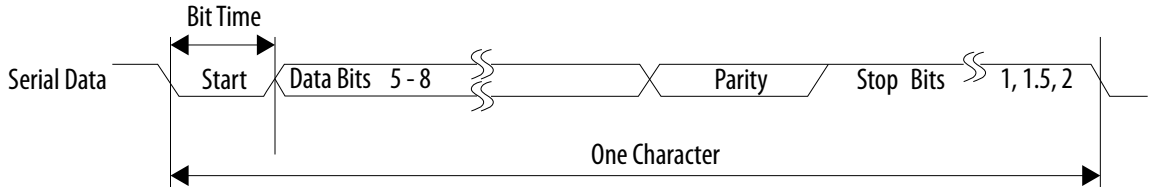
When FIFO buffer access mode is enabled, you can write data to the transmit FIFO buffer as normal; however, no serial transmission occurs in this mode and no data leaves the FIFO buffer. You can read back the data that is written to the transmit FIFO buffer with the transmit FIFO read (TFR) register. The TFR register provides the current data at the top of the transmit FIFO buffer. †

Similarly, you can also read data from the receive FIFO buffer in FIFO buffer access mode. Since the normal operation of the UART is halted in this mode, you must write data to the receive FIFO buffer to read it back. The receive FIFO write (RFW) register writes data to the receive FIFO buffer. The upper two bits of the 10-bit register write framing errors and parity error detection information to the receive FIFO buffer. Bit 9 of RFW indicates a framing error and bit 8 of RFW indicates a parity error. Although you cannot read these bits back from the receive buffer register, you can check the bits by reading the line status register (LSR), and by checking the corresponding bits when the data in question is at the top of the receive FIFO buffer. †

UART(RS232) Serial Protocol

Because the serial communication between the UART controller and the selected device is asynchronous, additional bits (start and stop) are added to the serial data to indicate the beginning and end. Utilizing these bits allows two devices to be synchronized. This structure of serial data accompanied by start and stop bits is referred to as a character, as shown in below.†

Figure 21-2: Serial Data Format



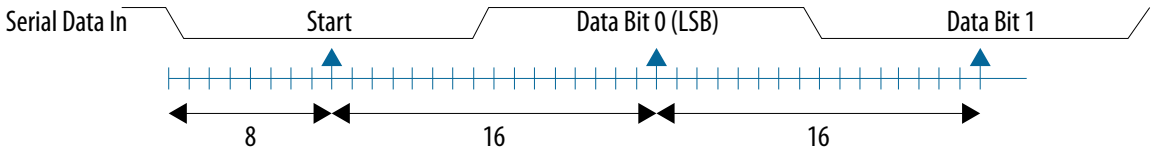
An additional parity bit may be added to the serial character. This bit appears after the last data bit and before the stop bit(s) in the character structure to provide the UART controller with the ability to perform simple error checking on the received data.†

The Control Register is used to control the serial character characteristics. The individual bits of the data word are sent after the start bit, starting with the least- significant bit (LSB). These are followed by the optional parity bit, followed by the stop bit(s), which can be 1, 1.5 or 2.†

All the bits in the transmission (with exception to the half stop bit when 1.5 stop bits are used) are transmitted for exactly the same time duration. This is referred to as a Bit Period or Bit Time. One Bit Time equals 16 baud clocks. To ensure stability on the line, the receiver samples the serial input data at approximately the midpoint of the Bit Time once the start bit has been detected. Because the exact number of baud clocks that each bit transmission is known, calculating the midpoint for sampling is not difficult. That is, every 16 baud clocks after the midpoint sample of the start bit.†

Together with serial input debouncing, this feature also contributes to avoid the detection of false start bits. Short glitches are filtered out by debouncing, and no transition is detected on the line. If a glitch is wide enough to avoid filtering by debouncing, a falling edge is detected. However, a start bit is detected only if the line is sampled low again after half a bit time has elapsed. †

Figure 21-3: Receiver Serial Data Sample Points



The baud rate of the UART controller is controlled by the serial clock and the Divisor Latch Register (DLH and DLL).†

Automatic Flow Control

The UART includes 16750-compatible request-to-send (RTS) and clear-to-send (CTS) serial data automatic flow control mode. You enable automatic flow control with the modem control register (`MCR.AFCE`). †

RTC Flow Control Trigger

RTC is an RX FIFO Almost-Full Trigger, where "almost full" refer to two available slots in the FIFO.

The UART controller uses two separate trigger levels for a DMA request and handshake signal (rts_n) in order to maximize throughput on the interface.

Automatic RTS mode

Automatic RTS mode becomes active when the following conditions occur: †

- RTS (`MCR.RTS` bit and `MCR.AFCE` bit are both set)
- FIFO buffers are enabled (`FCR.FIFOE` bit is set)

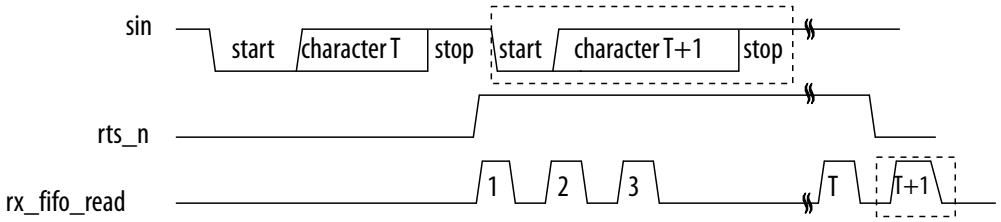
With automatic RTS enabled, the `rts_n` output pin is forced inactive (high) when the FIFO is almost full; where "almost full" refers to two available slots in the FIFO. When `rts_n` is connected to the `cts_n` input pin of another UART device, the other UART stops sending serial data until the receive FIFO buffer has available space (until it is completely empty). †

The selectable receive FIFO buffer threshold values are 1, ¼, ½, and 2 less than full. Because one additional character may be transmitted to the UART after `rts_n` is inactive (due to data already having entered the transmitter block in the other UART), setting the threshold to 2 less than full allows maximum use of the FIFO buffer with a margin of one character. †

Once the receive FIFO buffer is completely emptied by reading the receiver buffer register (`RBR_THR_DLL`), `rts_n` again becomes active (low), signaling the other UART to continue sending data.†

Even when you set the correct `MCR` bits, if the FIFO buffers are disabled through `FCR.FIFOE`, automatic flow control is also disabled. When auto RTS is not implemented or disabled, `rts_n` is controlled solely by `MCR.RTS`. In the Automatic RTS Timing diagram, the character T is received because `rts_n` is not detected prior to the next character entering the sending UART transmitter. †

Figure 21-4: Automatic RTS Timing



Automatic CTS mode

Automatic CTS mode becomes active when the following conditions occur: †

- AFCE (`MCR.AFCE` bit is set)
- FIFO buffers are enabled (through FIFO buffer control register `IIR_FCR.FIFOE`) bit

When automatic CTS is enabled (active), the UART transmitter is disabled whenever the `cts_n` input becomes inactive (high). This prevents overflowing the FIFO buffer of the receiving UART. †

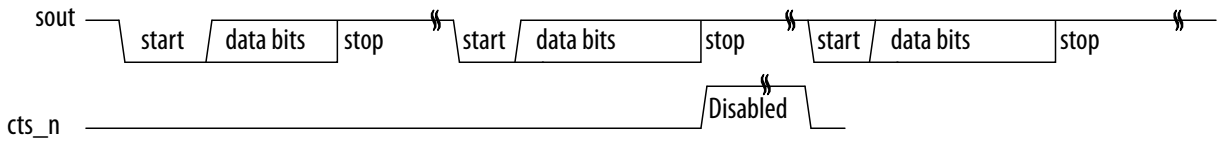
If the `cts_n` input is not deactivated before the middle of the last stop bit, another character is transmitted before the transmitter is disabled. While the transmitter is disabled, you can continue to write and even overflow to the transmit FIFO buffer. †

Automatic CTS mode requires the following sequence:

1. The UART status register are read to verify that the transmit FIFO buffer is full (UART status register `USR.TFNF` set to zero). †
2. The current FIFO buffer level is read via the transmit FIFO level (`TFL`) register. †
3. Programmable THRE interrupt mode must be enabled to access the FIFO buffer full status from the LSR. †

When using the FIFO buffer full status, software can poll this before each write to the transmit FIFO buffer. When the `cts_n` input becomes active (low) again, transmission resumes. If the FIFO buffers are disabled with the `FCR.FIFOE` bit, automatic flow control is also disabled regardless of any other settings. When auto CTS is not implemented or disabled, the transmitter is unaffected by `cts_n`.†

Figure 21-5: Automatic CTS Timing



Clocks

The UART controller is connected to the `l4_sp_clk` clock. The clock input is driven by the clock manager.

Related Information

Clock Manager on page 2-1
For more information, refer to the *Clock Manager* chapter.

Resets

The UART controller is connected to the `uart_rst_n` reset signal. The reset manager drives the signal on a cold or warm reset.

Taking the UART Controller Out of Reset

When a cold or warm reset is issued in the HPS, the reset manager resets this module and holds it in reset until software releases it.

After the Cortex-A9 MPCore CPU boots, it can deassert the reset signal by clearing the appropriate bits in the reset manager's corresponding reset register. For details about reset registers, refer to "Module Reset Signals".

Related Information

Reset Manager on page 3-1
For more information, refer to the *Reset Manager* chapter.

Interrupts

The assertion of the UART interrupt output signal occurs when one of the following interrupt types are enabled and active: †

Table 21-4: Interrupt Types and Priority †

Interrupt Type	Priority	Source	Interrupt Reset Control
Receiver line status	Highest	Overrun, parity and framing errors, break condition.	Reading the line status Register.
Received data available	Second	Receiver data available (FIFOs disabled) or RCVR FIFO trigger level reached (FIFOs enabled).	Reading the receiver buffer register (FIFOs disabled) or the FIFO drops below the trigger level (FIFOs enabled)
Character timeout indication	Second	No characters in or out of the Receive FIFO during the last 4 character times and there is at least 1 character in it during this Time.	Reading the receiver buffer Register.
Transmit holding register empty	Third	Transmitter holding register empty (Programmable THRE Mode disabled) or Transmit FIFO at or below threshold (Programmable THRE Mode enabled).	Reading the `IIR` register (if source of interrupt); or, writing into `THR` (FIFOs or Programmable THRE Mode not enabled) or Transmit FIFO above threshold (FIFOs and Programmable THRE Mode enabled).

Interrupt Type	Priority	Source	Interrupt Reset Control
Modem Status	Fourth	Clear to send or data set ready or ring indicator or data carrier detect. If auto flow control mode is enabled, a change in CTS (that is, DCTS set) does not cause an interrupt.	Reading the Modem status Register.

You can enable the interrupt types with the interrupt enable register (`IER_DLH`).

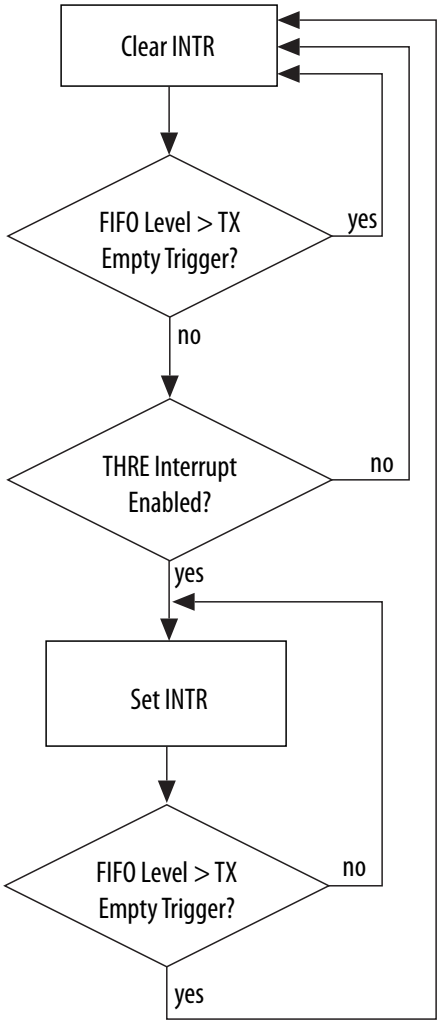
Note: Received Data Available" and "Character Timeout Indication" are enabled by a single bit in the IER_DLH register, because they have the same priority.

Once an interrupt is signaled, you can determine the interrupt source by reading the Interrupt Identity Register (IIR).

Programmable THRE Interrupt

The UART has a programmable THRE interrupt mode to increase system performance. You enable the programmable THRE interrupt mode with the interrupt enable register (`IER_DLH.PTIME`). When the THRE mode is enabled, THRE interrupts and the `dma_tx_req` signal are active at and below a programmed transmit FIFO buffer empty threshold level, as shown in the flowchart. †

Figure 21-6: Programmable THRE Interrupt

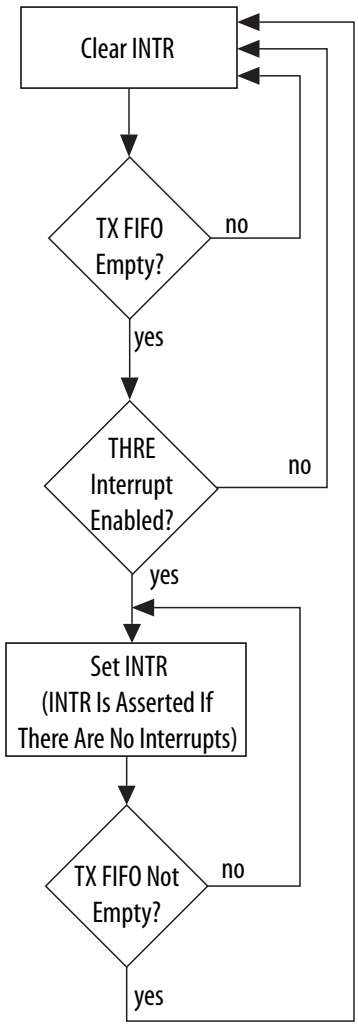


The threshold level is programmed into FCR.TET. The available empty thresholds are empty, 2, ¼, and ½. The optimum threshold value depends on the system's ability to begin a new transmission sequence in a timely manner. However, one of these thresholds should prove optimum in increasing system performance by preventing the transmit FIFO buffer from running empty.

In addition to the interrupt change, line status register (LSR.THRE) also switches from indicating that the transmit FIFO buffer is empty, to indicating that the FIFO buffer is full. This change allows software to fill the FIFO buffer for each transmit sequence by polling LSR.THRE before writing another character. This directs the UART to fill the transmit FIFO buffer whenever an interrupt occurs and there is data to transmit, instead of waiting until the FIFO buffer is completely empty. Waiting until the FIFO buffer is empty reduces performance whenever the system is too busy to respond immediately. You can increase system efficiency when this mode is enabled in combination with automatic flow control.

When not selected or disabled, THRE interrupts and LSR.THRE function normally, reflecting an empty THR or FIFO buffer.

Figure 21-7: Interrupt Generation without Programmable THRE Interrupt Mode



DMA Controller Operation

The UART controller includes a DMA controller interface to indicate when the receive FIFO buffer data is available or when the transmit FIFO buffer requires data. The DMA requires two channels, one for transmit and one for receive. The UART controller supports both single and burst transfers.

The FIFO buffer depth (`FIFO_DEPTH`) for both the RX and TX buffers in the UART controller is 128 entries.

Related Information

DMA Controller on page 16-1
For more information, refer to the DMA Controller chapter.

Transmit FIFO Underflow

During UART serial transfers, transmit FIFO requests are made to the DMA controller whenever the number of entries in the transmit FIFO is less than or equal to the decoded level of the Transmit Empty Trigger (TET) field in the FIFO Control Register (FCR), also known as the watermark level. The DMA controller responds by writing a burst of data to the transmit FIFO buffer, of length specified as DMA burst length. †

Data should be fetched from the DMA often enough for the transmit FIFO to perform serial transfers continuously, that is, when the FIFO begins to empty, another DMA request should be triggered. Otherwise, the FIFO will run out of data (underflow) causing a STOP to be inserted on the UART bus. To prevent this condition, you must set the watermark level correctly. †

Related Information

DMA Controller on page 16-1
For more information, refer to the DMA Controller chapter.

Transmit Watermark Level

Consider the example where the following assumption is made: †

DMA burst length = FIFO_DEPTH - decoded watermark level of IIR_FCR.TET †

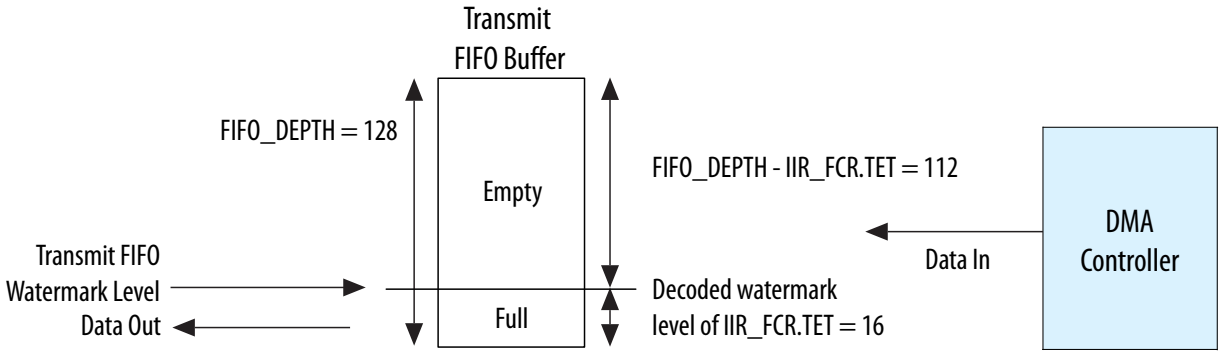
Here the number of data items to be transferred in a DMA burst is equal to the empty space in the transmit FIFO. Consider the following two different watermark level settings: †

IIR_FCR.TET = 1

IIR_FCR.TET = 1 decodes to a watermark level of 16.

- Transmit FIFO watermark level = decoded watermark level of IIR_FCR.TET = 16 †
- DMA burst length = FIFO_DEPTH - decoded watermark level of IIR_FCR.TET = 112 †
- UART transmit FIFO_DEPTH = 128 †
- Block transaction size = 448†

Figure 21-8: Transmit FIFO Watermark Level = 16



The number of burst transactions needed equals the block size divided by the number of data items per burst:

Block transaction size/DMA burst length = 448/112 = 4

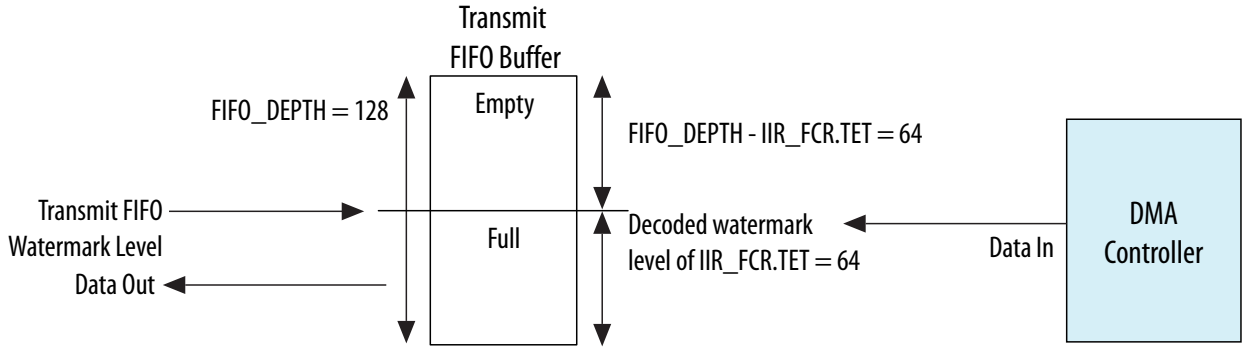
The number of burst transactions in the DMA block transfer is 4. But the watermark level, decoded level of IIR_FCR.TET, is quite low. Therefore, the probability of transmit underflow is high where the UART serial transmit line needs to transmit data, but there is no data left in the transmit FIFO. This occurs because the DMA has not had time to service the DMA request before the FIFO becomes empty.

IIR_FCR.TET = 3

IIR_FCR.TET = 3 decodes to a watermark level of 64.

- Transmit FIFO watermark level = decoded watermark level of IIR_FCR.TET = 64 †
- DMA burst length = FIFO_DEPTH - decoded watermark level of IIR_FCR.TET = 64†
- UART transmit FIFO_DEPTH = 128 †
- Block transaction size = 448 †

Figure 21-9: Transmit FIFO Watermark Level = 64



Number of burst transactions in block: †

Block transaction size/DMA burst length = 448/64 = 7 †

In this block transfer, there are 15 destination burst transactions in a DMA block transfer. But the watermark level, decoded level of IIR_FCR.TET, is high. Therefore, the probability of UART transmit underflow is low because the DMA controller has plenty of time to service the destination burst transaction request before the UART transmit FIFO becomes empty. †

Thus, the second case has a lower probability of underflow at the expense of more burst transactions per block. This provides a potentially greater amount of bursts per block and worse bus utilization than the former case. †

Therefore, the goal in choosing a watermark level is to minimize the number of transactions per block, while at the same time keeping the probability of an underflow condition to an acceptable level. In practice, this is a function of the ratio of the rate at which the UART transmits data to the rate at which the DMA can respond to destination burst requests. †

Transmit FIFO Overflow

Setting the DMA burst length to a value greater than the watermark level that triggers the DMA request might cause overflow when there is not enough space in the transmit FIFO to service the destination burst request. Therefore, the following equation must be adhered to in order to avoid overflow: †

DMA burst length <= FIFO_DEPTH - decoded watermark level of IIR_FCR.TET

In case 2: decoded watermark level of IIR_FCR.TET = 64, the amount of space in the transmit FIFO at the time of the burst request is made is equal to the DMA burst length. Thus, the transmit FIFO may be full, but not overflowed, at the completion of the burst transaction. †

Therefore, for optimal operation, DMA burst length must be set at the FIFO level that triggers a transmit DMA request; that is: †

DMA burst length = FIFO_DEPTH - decoded watermark level of IIR_FCR.TET

Adhering to this equation reduces the number of DMA bursts needed for block transfer, and this in turn improves bus utilization. †

The transmit FIFO will not be full at the end of a DMA burst transfer if the UART controller has successfully transmitted one data item or more on the UART serial transmit line during the transfer. †

Receive FIFO Overflow

During UART serial transfers, receive FIFO requests are made to the DMA whenever the number of entries in the receive FIFO is at or above the decoded level of Receive Trigger (RT) field in the FIFO Control Register (IIR_FCR). This is known as the watermark level. The DMA responds by fetching a burst of data from the receive FIFO. †

Data should be fetched by the DMA often enough for the receive FIFO to accept serial transfers continuously, that is, when the FIFO begins to fill, another DMA transfer is requested. Otherwise the FIFO will fill with data (overflow). To prevent this condition, the user must set the watermark level correctly. †

Receive Watermark Level

Similar to choosing the transmit watermark level described earlier, the receive watermark level, decoded watermark level of IIR_FCR.RT, should be set to minimize the probability of overflow, as shown in the Receive FIFO Buffer diagram. It is a tradeoff between the number of DMA burst transactions required per block versus the probability of an overflow occurring. †

Receive FIFO Underflow

Setting the source transaction burst length greater than the watermark level can cause underflow where there is not enough data to service the source burst request. Therefore, the following equation must be adhered to avoid underflow: †

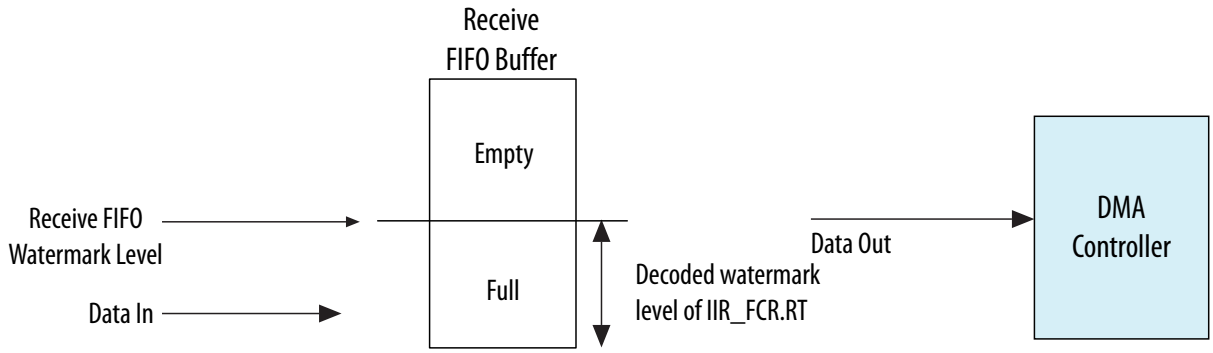
DMA burst length = decoded watermark level of IIR_FCR.RT + 1

If the number of data items in the receive FIFO is equal to the source burst length at the time of the burst request is made, the receive FIFO may be emptied, but not underflowed, at the completion of the burst transaction. For optimal operation, DMA burst length should be set at the watermark level, decoded watermark level of IIR_FCR.RT. †

Adhering to this equation reduces the number of DMA bursts in a block transfer, which in turn can avoid underflow and improve bus utilization. †

The receive FIFO will not be empty at the end of the source burst transaction if the UART controller has successfully received one data item or more on the UART serial receive line during the burst. †

Figure 21-10: Receive FIFO Buffer



UART Controller Address Map and Register Definitions

For complete HPS address map and register definitions, refer to the **Arria 10 HPS Address Map and Register Definitions**.

uart Address Map

Module Instance	Base Address	End Address
i_uart_0_uart	0xFFC02000	0xFFC020FF
i_uart_1_uart	0xFFC02100	0xFFC021FF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
rbr_thr_dll on page 21-34	0x0	32	RW	0x0	Rx Buffer, Tx Holding, and Divisor Latch Low
ier_dlh on page 21-36	0x4	32	RW	0x0	Interrupt Enable and Divisor Latch High
iir on page 21-38	0x8	32	RO	0x1	Interrupt Identification Register
fcr on page 21-40	0x8	32	WO	0x0	FIFO Control Register
lcr on page 21-44	0xC	32	RW	0x0	Line Control Register

Register	Offset	Width	Access	Reset Value	Description
mcr on page 21-48	0x10	32	RW	0x0	Modem Control Register
lsr on page 21-53	0x14	32	RO	0x60	Line Status Register
msr on page 21-59	0x18	32	RO	0x0	Modem Status Register It should be noted that whenever bits 0, 1, 2 or 3 is set to logic one, to indicate a change on the modem control inputs, a modem status interrupt will be generated if enabled via the IER regardless of when the change occurred. Since the delta bits (bits 0, 1, 3) can get set after a reset if their respective modem signals are active (see individual bits for details), a read of the MSR after reset can be performed to prevent unwanted interrupts.
scr on page 21-65	0x1C	32	RW	0x0	Scratchpad Register
srbr_sthr_0 on page 21-66	0x30	32	RW	0x0	Shadow RBR and THR
srbr_sthr_1 on page 21-69	0x34	32	RW	0x0	Shadow RBR and THR
srbr_sthr_2 on page 21-69	0x38	32	RW	0x0	Shadow RBR and THR
srbr_sthr_3 on page 21-70	0x3C	32	RW	0x0	Shadow RBR and THR
srbr_sthr_4 on page 21-71	0x40	32	RW	0x0	Shadow RBR and THR
srbr_sthr_5 on page 21-72	0x44	32	RW	0x0	Shadow RBR and THR
srbr_sthr_6 on page 21-73	0x48	32	RW	0x0	Shadow RBR and THR

Register	Offset	Width	Access	Reset Value	Description
srbr_sthr_7 on page 21-73	0x4C	32	RW	0x0	Shadow RBR and THR
srbr_sthr_8 on page 21-74	0x50	32	RW	0x0	Shadow RBR and THR
srbr_sthr_9 on page 21-75	0x54	32	RW	0x0	Shadow RBR and THR
srbr_sthr_10 on page 21-76	0x58	32	RW	0x0	Shadow RBR and THR
srbr_sthr_11 on page 21-77	0x5C	32	RW	0x0	Shadow RBR and THR
srbr_sthr_12 on page 21-77	0x60	32	RW	0x0	Shadow RBR and THR
srbr_sthr_13 on page 21-78	0x64	32	RW	0x0	Shadow RBR and THR
srbr_sthr_14 on page 21-79	0x68	32	RW	0x0	Shadow RBR and THR
srbr_sthr_15 on page 21-80	0x6C	32	RW	0x0	Shadow RBR and THR
far on page 21-81	0x70	32	RW	0x0	FIFO Access Register
tfr on page 21-82	0x74	32	RO	0x0	Transmit FIFO Read
rfw on page 21-83	0x78	32	RW	0x0	Receive FIFO Write
usr on page 21-85	0x7C	32	RO	0x6	UART Status register.
tfl on page 21-88	0x80	32	RO	0x0	
rfl on page 21-89	0x84	32	RO	0x0	Receive FIFO Level.
srr on page 21-90	0x88	32	RW	0x0	Software Reset Register.
srts on page 21-92	0x8C	32	RW	0x0	Shadow Request to Send.
sbcr on page 21-95	0x90	32	RW	0x0	Shadow Break Control Register.
sdmam on page 21-96	0x94	32	RW	0x0	Shadow DMA Mode.
sfe on page 21-97	0x98	32	RW	0x0	Shadow FIFO Enable

Register	Offset	Width	Access	Reset Value	Description
srt on page 21-99	0x9C	32	RW	0x0	Shadow RCVR Trigger
stet on page 21-100	0xA0	32	RW	0x0	Shadow TX Empty Trigger
htx on page 21-102	0xA4	32	RW	0x0	Halt TX
dmasa on page 21-103	0xA8	32	RW	0x0	DMA Software Acknowledge
cpr on page 21-104	0xF4	32	RO	0x83F32	Component Parameter Register
ucv on page 21-108	0xF8	32	RO	0x3331342A	Component Version
ctr on page 21-109	0xFC	32	RO	0x44570110	Component Type Register

uart Summary

Module Instance	Base Address
i_uart_0_uart	0xFFC02000
i_uart_1_uart	0xFFC02100

Register Address Offset	Bit Fields															
i_uart_0_uart																
rbr_thr_dll 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	rsvd_rbr_thr_dll_31to8 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_rbr_thr_dll_31to8 RO 0x0								value RW 0x0							

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	rsvd_ier_dlh_31to8 RO 0x0															
ier_dlh 0x4	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_ier_dlh_31to8 RO 0x0								ptime_dlh7 RW 0x0	dlh6 RW 0x0	dlh5 RW 0x0	dlh4 RW 0x0	edssi_dhl3 RW 0x0	elsi_dhl2 RW 0x0	etbei_dlhl RW 0x0	erbfi_dlh0 RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	rsvd_iir_31to8 RO 0x0															
iir 0x8	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_iir_31to8 RO 0x0								fifoen RO 0x0		rsvd_iir_5to4 RO 0x0		id RO 0x1			
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
fcr 0x8	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								rt WO 0x0		tet WO 0x0		dmam WO 0x0	xfifor WO 0x0	rfifor WO 0x0	fifoe WO 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	rsvd_lcr_31to8 RO 0x0															
lcr 0xC	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_lcr_31to8 RO 0x0								dlab RW 0x0	break RW 0x0	sp RW 0x0	eps RW 0x0	pen RW 0x0	stop RW 0x0	dls RW 0x0	
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	rsvd_mcr_31to7 RO 0x0															
mcr 0x10	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_mcr_31to7 RO 0x0									sire RO 0x0	afce RW 0x0	loopback RW 0x0	out2 RW 0x0	out1 RW 0x0	rts RW 0x0	dtr RW 0x0

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
lsr 0x14	rsvd_lsr_31to8 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_lsr_31to8 RO 0x0								rfe RO 0x0	temt RO 0x1	thre RO 0x1	bi RO 0x0	fe RO 0x0	pe RO 0x0	oe RO 0x0	dr RO 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
msr 0x18	rsvd_msc_31to8 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_msc_31to8 RO 0x0								dcd RO 0x0	ri RO 0x0	dsr RO 0x0	cts RO 0x0	ddcd RO 0x0	teri RO 0x0	ddsr RO 0x0	dcts RO 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
scr 0x1C	rsvd_scr_31to8 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_scr_31to8 RO 0x0								scr RW 0x0							
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
srbr_sthr_0 0x30	rsvd_srbr_sthr_0_31to8 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_srbr_sthr_0_31to8 RO 0x0								srbr_sthr_0 RO 0x0							
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
srbr_sthr_1 0x34	rsvd_srbr_sthr_1_31to8 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_srbr_sthr_1_31to8 RO 0x0								srbr_sthr_1 RO 0x0							
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
srbr_sthr_2 0x38	rsvd_srbr_sthr_2_31to8 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_srbr_sthr_2_31to8 RO 0x0								srbr_sthr_2 RO 0x0							

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
srbr_sthr_3	rsvd_srbr_sthr_3_31to8 RO 0x0															
0x3C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_srbr_sthr_3_31to8 RO 0x0								srbr_sthr_3 RO 0x0							
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
srbr_sthr_4	rsvd_srbr_sthr_4_31to8 RO 0x0															
0x40	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_srbr_sthr_4_31to8 RO 0x0								srbr_sthr_4 RO 0x0							
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
srbr_sthr_5	rsvd_srbr_sthr_5_31to8 RO 0x0															
0x44	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_srbr_sthr_5_31to8 RO 0x0								srbr_sthr_5 RO 0x0							
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
srbr_sthr_6	rsvd_srbr_sthr_6_31to8 RO 0x0															
0x48	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_srbr_sthr_6_31to8 RO 0x0								srbr_sthr_6 RO 0x0							
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
srbr_sthr_7	rsvd_srbr_sthr_7_31to8 RO 0x0															
0x4C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_srbr_sthr_7_31to8 RO 0x0								srbr_sthr_7 RO 0x0							
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
srbr_sthr_8	rsvd_srbr_sthr_8_31to8 RO 0x0															
0x50	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_srbr_sthr_8_31to8 RO 0x0								srbr_sthr_8 RO 0x0							

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
srbr_sthr_9	rsvd_srbr_sthr_9_31to8 RO 0x0															
0x54	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_srbr_sthr_9_31to8 RO 0x0								srbr_sthr_9 RO 0x0							
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
srbr_sthr_10	rsvd_srbr_sthr_10_31to8 RO 0x0															
0x58	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_srbr_sthr_10_31to8 RO 0x0								srbr_sthr_10 RO 0x0							
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
srbr_sthr_11	rsvd_srbr_sthr_11_31to8 RO 0x0															
0x5C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_srbr_sthr_11_31to8 RO 0x0								srbr_sthr_11 RO 0x0							
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
srbr_sthr_12	rsvd_srbr_sthr_12_31to8 RO 0x0															
0x60	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_srbr_sthr_12_31to8 RO 0x0								srbr_sthr_12 RO 0x0							
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
srbr_sthr_13	rsvd_srbr_sthr_13_31to8 RO 0x0															
0x64	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_srbr_sthr_13_31to8 RO 0x0								srbr_sthr_13 RO 0x0							
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
srbr_sthr_14	rsvd_srbr_sthr_14_31to8 RO 0x0															
0x68	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_srbr_sthr_14_31to8 RO 0x0								srbr_sthr_14 RO 0x0							

Register Address Offset	Bit Fields															
srbr_sthr_15 0x6C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	rsvd_srbr_sthr_15_31to8 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_srbr_sthr_15_31to8 RO 0x0								srbr_sthr_15 RO 0x0							
far 0x70	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	rsvd_far_31to1 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_far_31to1 RO 0x0															srbr_sthr RW 0x0
tfr 0x74	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	rsvd_tfr_31to8 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_tfr_31to8 RO 0x0								tfr RO 0x0							
rfw 0x78	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	rsvd_rfw_31to10 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_rfw_31to10 RO 0x0						rffe WO 0x0	rfpe WO 0x0	rfwd WO 0x0							
usr 0x7C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	rsvd_usr_31to5 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_usr_31to5 RO 0x0											rff RO 0x0	rfne RO 0x0	tfe RO 0x1	tfnf RO 0x1	rsvd_busy RO 0x0

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
tfl	rsvd_tfl_31toaddr_width RO 0x0															
0x80	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_tfl_31toaddr_width RO 0x0								tfl RO 0x0							
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
rfl	rsvd_rfl_31toaddr_width RO 0x0															
0x84	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_rfl_31toaddr_width RO 0x0								rfl RO 0x0							
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
srr	rsvd_srr_31to3 RO 0x0															
0x88	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_srr_31to3 RO 0x0													xfr WO 0x0	rfr WO 0x0	ur WO 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
srts	rsvd_srts_31to1 RO 0x0															
0x8C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_srts_31to1 RO 0x0															srts RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
sbcr	rsvd_sbcr_31to1 RO 0x0															
0x90	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_sbcr_31to1 RO 0x0															sbcr RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
sdmam	rsvd_sdmam_31to1 RO 0x0															
0x94	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_sdmam_31to1 RO 0x0															sdmam RW 0x0

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
sfe	rsvd_sfe_31to1 RO 0x0															
0x98	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_sfe_31to1 RO 0x0															sfe RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
srt	rsvd_srt_31to2 RO 0x0															
0x9C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_srt_31to2 RO 0x0														srt RW 0x0	
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
stet	rsvd_stet_31to2 RO 0x0															
0xA0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_stet_31to2 RO 0x0														stet RW 0x0	
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
htx	rsvd_htx_31to1 RO 0x0															
0xA4	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_htx_31to1 RO 0x0															htx RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dmasa	rsvd_dmasa_31to1 RO 0x0															
0xA8	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_dmasa_31to1 RO 0x0															dmasa WO 0x0

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	rsvd_cpr_31to24 RO 0x0								fifo_mode RO 0x8							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cpr 0xF4	rsvd_cpr_15to14 RO 0x0		dma_extra RO 0x1	uart_add_encoded_param RO 0x1	shadow RO 0x1	fifo_stat RO 0x1	fifo_access RO 0x1	additional_feat RO 0x1	sir_lp_mode RO 0x0	sir_mode RO 0x0	thre_mode RO 0x1	afce_mode RO 0x1	rsvd_cpr_3to2 RO 0x0		apbdatawidth RO 0x2	
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	uart_component_version RO 0x3331342A															
ucv 0xF8	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	uart_component_version RO 0x3331342A															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	peripheral_id RO 0x44570110															
ctr 0xFC	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	peripheral_id RO 0x44570110															
i_uart_1_uart																
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	rsvd_rbr_thr_dll_31to8 RO 0x0															
rbr_thr_dll 0x0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_rbr_thr_dll_31to8 RO 0x0								value RW 0x0							
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	rsvd_ier_dlh_31to8 RO 0x0															
ier_dlh 0x4	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_ier_dlh_31to8 RO 0x0								ptime_dlh7 RW 0x0	dlh6 RW 0x0	dlh5 RW 0x0	dlh4 RW 0x0	edssi_dhl3 RW 0x0	elsi_dhl2 RW 0x0	etbei_dlhl RW 0x0	erbfi_dlh0 RW 0x0

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	rsvd_iir_31to8 RO 0x0															
iir	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x8	rsvd_iir_31to8 RO 0x0								fifoen RO 0x0		rsvd_iir_5to4 RO 0x0		id RO 0x1			
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
fcr	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x8	Reserved								rt WO 0x0		tet WO 0x0		dmam WO 0x0	xfifor WO 0x0	rfifor WO 0x0	fifoe WO 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	rsvd_lcr_31to8 RO 0x0															
lcr	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xC	rsvd_lcr_31to8 RO 0x0								dlab RW 0x0	break RW 0x0	sp RW 0x0	eps RW 0x0	pen RW 0x0	stop RW 0x0	dls RW 0x0	
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	rsvd_mcr_31to7 RO 0x0															
mcr	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x10	rsvd_mcr_31to7 RO 0x0									sire RO 0x0	afce RW 0x0	loopback RW 0x0	out2 RW 0x0	out1 RW 0x0	rts RW 0x0	dtr RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	rsvd_lsr_31to8 RO 0x0															
lsr	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x14	rsvd_lsr_31to8 RO 0x0								rfe RO 0x0	temt RO 0x1	thre RO 0x1	bi RO 0x0	fe RO 0x0	pe RO 0x0	oe RO 0x0	dr RO 0x0

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	rsvd_msc_31to8 RO 0x0															
msr 0x18	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_msc_31to8 RO 0x0								dcd RO 0x0	ri RO 0x0	dsr RO 0x0	cts RO 0x0	ddcd RO 0x0	teri RO 0x0	ddsr RO 0x0	dcts RO 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	rsvd_scr_31to8 RO 0x0															
scr 0x1C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_scr_31to8 RO 0x0								scr RW 0x0							
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	rsvd_srbr_sthr_0_31to8 RO 0x0															
srbr_sthr_0 0x30	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_srbr_sthr_0_31to8 RO 0x0								srbr_sthr_0 RO 0x0							
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	rsvd_srbr_sthr_1_31to8 RO 0x0															
srbr_sthr_1 0x34	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_srbr_sthr_1_31to8 RO 0x0								srbr_sthr_1 RO 0x0							
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	rsvd_srbr_sthr_2_31to8 RO 0x0															
srbr_sthr_2 0x38	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_srbr_sthr_2_31to8 RO 0x0								srbr_sthr_2 RO 0x0							
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	rsvd_srbr_sthr_3_31to8 RO 0x0															
srbr_sthr_3 0x3C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_srbr_sthr_3_31to8 RO 0x0								srbr_sthr_3 RO 0x0							

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
srbr_sthr_4	rsvd_srbr_sthr_4_31to8 RO 0x0															
0x40	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_srbr_sthr_4_31to8 RO 0x0								srbr_sthr_4 RO 0x0							
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
srbr_sthr_5	rsvd_srbr_sthr_5_31to8 RO 0x0															
0x44	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_srbr_sthr_5_31to8 RO 0x0								srbr_sthr_5 RO 0x0							
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
srbr_sthr_6	rsvd_srbr_sthr_6_31to8 RO 0x0															
0x48	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_srbr_sthr_6_31to8 RO 0x0								srbr_sthr_6 RO 0x0							
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
srbr_sthr_7	rsvd_srbr_sthr_7_31to8 RO 0x0															
0x4C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_srbr_sthr_7_31to8 RO 0x0								srbr_sthr_7 RO 0x0							
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
srbr_sthr_8	rsvd_srbr_sthr_8_31to8 RO 0x0															
0x50	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_srbr_sthr_8_31to8 RO 0x0								srbr_sthr_8 RO 0x0							
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
srbr_sthr_9	rsvd_srbr_sthr_9_31to8 RO 0x0															
0x54	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_srbr_sthr_9_31to8 RO 0x0								srbr_sthr_9 RO 0x0							

Register Address Offset	Bit Fields															
srbr_sthr_10 0x58	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	rsvd_srbr_sthr_10_31to8 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_srbr_sthr_10_31to8 RO 0x0								srbr_sthr_10 RO 0x0							
srbr_sthr_11 0x5C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	rsvd_srbr_sthr_11_31to8 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_srbr_sthr_11_31to8 RO 0x0								srbr_sthr_11 RO 0x0							
srbr_sthr_12 0x60	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	rsvd_srbr_sthr_12_31to8 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_srbr_sthr_12_31to8 RO 0x0								srbr_sthr_12 RO 0x0							
srbr_sthr_13 0x64	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	rsvd_srbr_sthr_13_31to8 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_srbr_sthr_13_31to8 RO 0x0								srbr_sthr_13 RO 0x0							
srbr_sthr_14 0x68	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	rsvd_srbr_sthr_14_31to8 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_srbr_sthr_14_31to8 RO 0x0								srbr_sthr_14 RO 0x0							
srbr_sthr_15 0x6C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	rsvd_srbr_sthr_15_31to8 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_srbr_sthr_15_31to8 RO 0x0								srbr_sthr_15 RO 0x0							

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
far 0x70	rsvd_far_31to1 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_far_31to1 RO 0x0															srbr_sthr RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
tfr 0x74	rsvd_tfr_31to8 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_tfr_31to8 RO 0x0								tfr RO 0x0							
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
rfw 0x78	rsvd_rfw_31to10 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_rfw_31to10 RO 0x0						rffe WO 0x0	rfpe WO 0x0	rfwd WO 0x0							
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
usr 0x7C	rsvd_usr_31to5 RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_usr_31to5 RO 0x0											rff RO 0x0	rfne RO 0x0	tfe RO 0x1	tfnf RO 0x1	rsvd_busy RO 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
tfl 0x80	rsvd_tfl_31toaddr_width RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_tfl_31toaddr_width RO 0x0								tfl RO 0x0							

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
rfl	rsvd_rfl_31toaddr_width RO 0x0															
0x84	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_rfl_31toaddr_width RO 0x0								rfl RO 0x0							
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
srr	rsvd_srr_31to3 RO 0x0															
0x88	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_srr_31to3 RO 0x0													xfr WO 0x0	rfr WO 0x0	ur WO 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
srts	rsvd_srts_31to1 RO 0x0															
0x8C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_srts_31to1 RO 0x0															srts RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
sbcr	rsvd_sbcr_31to1 RO 0x0															
0x90	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_sbcr_31to1 RO 0x0															sbcr RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
sdmam	rsvd_sdmam_31to1 RO 0x0															
0x94	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_sdmam_31to1 RO 0x0															sdmam RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
sfe	rsvd_sfe_31to1 RO 0x0															
0x98	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_sfe_31to1 RO 0x0															sfe RW 0x0

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
srt	rsvd_srt_31to2 RO 0x0															
0x9C	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_srt_31to2 RO 0x0														srt RW 0x0	
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
stet	rsvd_stet_31to2 RO 0x0															
0xA0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_stet_31to2 RO 0x0														stet RW 0x0	
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
htx	rsvd_htx_31to1 RO 0x0															
0xA4	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_htx_31to1 RO 0x0															htx RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dmasa	rsvd_dmasa_31to1 RO 0x0															
0xA8	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_dmasa_31to1 RO 0x0															dmasa WO 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	rsvd_cpr_31to24 RO 0x0								fifo_mode RO 0x8							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cpr	rsvd_cpr_15to14 RO 0x0		dma_extra RO 0x1	uart_add_encoded_param RO 0x1	shadow RO 0x1	fifo_stat RO 0x1	fifo_access RO 0x1	additional_feat RO 0x1	sir_lp_mode RO 0x0	sir_mode RO 0x0	thre_mode RO 0x1	afce_mode RO 0x1	rsvd_cpr_3to2 RO 0x0		apbdatawidth RO 0x2	
0xF4																

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ucv 0xF8	uart_component_version RO 0x3331342A															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	uart_component_version RO 0x3331342A															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ctr 0xFC	peripheral_id RO 0x44570110															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	peripheral_id RO 0x44570110															

rbr_thr_dll

This is a multi-function register. This register holds receives and transmit data and controls the least-signficant 8 bits of the baud rate divisor.

Module Instance	Base Address	Register Address
i_uart_0_uart	0xFFC02000	0xFFC02000
i_uart_1_uart	0xFFC02100	0xFFC02100

Offset: 0x0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
rsvd_rbr_thr_dll_31to8 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rsvd_rbr_thr_dll_31to8 RO 0x0								value RW 0x0							

rbr_thr_dll Fields

Bit	Name	Description	Access	Reset
31:8	rsvd_rbr_thr_dll_31to8	Reserved bits [31:8] - Read Only	RO	0x0
7:0	value	Receive Buffer Register: This register contains the data byte received on the serial input port (uart_rxd). The data in this register is valid only if the Data Ready (bit [0] in the Line Status Register(LSR)) is set to 1. If FIFOs are disabled(bit[0] of Register FCR is set to 0) the data in the RBR must be read before the next data arrives, otherwise it will be overwritten, resulting in an overrun error. If FIFOs are enabled(bit [0] of Register FCR is set to 1) this register accesses the head of the receive FIFO. If the receive FIFO is full, and this register is not read before the next data character arrives, then the data already in the FIFO will be preserved but any incoming data will be lost. An overrun error will also occur. Transmit Holding Register: This register contains data to be transmitted on the serial output port. Data should only be written to the THR when the THR Empty bit [5] of the LSR Register is set to 1. If FIFOs are disabled (bit [0] of Register FCR) is set to 0 and THRE is set to 1, writing a single character to the THR clears the THRE. Any additional writes to the THR before the THRE is set again causes the THR data to be overwritten. If FIFO's are enabled bit [0] of Register FCR is set to 1 and THRE is set up to 128 characters of data may be written to the THR before the FIFO is full. Any attempt to write data when the FIFO is full results in the write data being lost. Divisor Latch Low: This register makes up the lower 8-bits of a 16-bit, Read/write, Divisor Latch register that contains the baud rate divisor for the UART. This register may only be accessed when the DLAB bit [7] of the LCR Register is set to 1. The output baud rate is equal to the serial clock l4_sp_clk frequency divided by sixteen times the value of the baud rate divisor, as follows: baud rate = (serial clock freq) / (16 * divisor) Note that with the Divisor Latch Registers (DLL and DLH) set to zero, the baud clock is disabled and no serial communications will occur. Also, once the DLL is set, at least 8 l4_sp_clk clock cycles should be allowed to pass before transmitting or receiving data.	RW	0x0

ier_dlh

This is a multi-function register. This register enables/ disables receive and transmit interrupts and also controls the most-significant 8-bits of the baud rate divisor.
Divisor Latch High Register:
This register is accessed when the DLAB bit [7] of the LCR Register is set to 1.Bits[7:0] contain the high order 8-bits of the baud rate divisor.The output baud rate is equal to the serial clock l4_sp_clk frequency divided by sixteen times the value of the baud rate divisor, as follows:
baud rate = (serial clock freq) / (16 * divisor):
Note that with the Divisor Latch Registers (DLLand DLH) set to zero, the baud clock is disabled and no serial communications will occur. Also, once the DLL is set, at least 8 l4_sp_clk clock cycles should be allowed to pass before transmitting or receiving data.
Interrupt Enable Register:
This register may only be accessed when the DLAB bit [7] of the LCR Register is set to 0.Allows control of the Interrupt Enables for transmit and receive functions.

Module Instance	Base Address	Register Address
i_uart_0_uart	0xFFC02000	0xFFC02004
i_uart_1_uart	0xFFC02100	0xFFC02104

Offset: 0x4

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
rsvd_ier_dlh_31to8 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rsvd_ier_dlh_31to8 RO 0x0								ptime_dlh7 RW 0x0	dlh6 RW 0x0	dlh5 RW 0x0	dlh4 RW 0x0	edssi_dhl3 RW 0x0	elsi_dhl2 RW 0x0	etbei_dlhl RW 0x0	erbfi_dlh0 RW 0x0

ier_dlh Fields

Bit	Name	Description	Access	Reset
31:8	rsvd_ier_dlh_31to8	Reserved bits [31:8] - Read Only	RO	0x0
7	ptime_dlh7	Divisor Latch High Register: Bit 7 of DLH value. Interrupt Enable Register: This is used to enable/disable the generation of THRE Interrupt. **Value** **Description** 0 DISABLED 1 ENABLED	RW	0x0
6	dlh6	Bit 6 of DLH value.	RW	0x0
5	dlh5	Bit 5 of DLH value.	RW	0x0
4	dlh4	Bit 4 of DLH value.	RW	0x0
3	edssi_dhl3	Divisor Latch High Register: Bit 3 of DLH value. Interrupt Enable Register: This is used to enable/disable the generation of Modem Status Interrupts. This is the fourth highest priority interrupt. **Value** **Description** 0 DISABLED 1 ENABLED	RW	0x0
2	elsi_dhl2	Divisor Latch High Register: Bit 2 of DLH value. Interrupt Enable Register: This is used to enable/disable the generation of Receiver Line Status Interrupt. This is the highest priority interrupt. **Value** **Description** 0 DISABLED 1 ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
1	etbei_dlhl	Divisor Latch High Register: Bit 1 of DLH value. Interrupt Enable Register: Enable Transmit Holding Register Empty Interrupt. This is used to enable/disable the generation of Transmitter Holding Register Empty Interrupt. This is the third highest priority interrupt. **Value** **Description** 0 DISABLED 1 ENABLED	RW	0x0
0	erbfi_dlh0	Divisor Latch High Register: Bit 0 of DLH value. Interrupt Enable Register: Used to enable/disable the generation of the Receive Data Available Interrupt and the Character Timeout Interrupt(if FIFO's enabled). These are the second highest priority interrupts. **Value** **Description** 0 DISABLED 1 ENABLED	RW	0x0

iir

Interrupt Identification Register

Module Instance	Base Address	Register Address
i_uart_0_uart	0xFFC02000	0xFFC02008
i_uart_1_uart	0xFFC02100	0xFFC02108

Offset: 0x8

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
rsvd_iir_31to8 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rsvd_iir_31to8 RO 0x0								fifoen RO 0x0		rsvd_iir_5to4 RO 0x0		id RO 0x1			

iir Fields

Bit	Name	Description	Access	Reset
31:8	rsvd_iir_31to8	Reserved bits [31:8] - Read Only	RO	0x0
7:6	fifoen	Bits[7:6], FIFO's Enabled (or FIFOSE): This is used to indicate whether the FIFO's are enabled or disabled. 00 = disabled. 11 = enabled **Value** **Description** 0 DISABLED 3 ENABLED	RO	0x0
5:4	rsvd_iir_5to4	Reserved bits [5:4] - Read Only	RO	0x0

Bit	Name	Description	Access	Reset
3:0	id	Bits[3:0], Interrupt ID (or IID): This indicates the highest priority pending interrupt which can be one of the following types: 0000 = modem status. 0001 = no interrupt pending. 0010 = THR empty. 0100 = received data available. 0110 = receiver line status. 0111 = busy detect. 1100 = character timeout. Note, an interrupt of type 0111 (busy detect) will never get indicated if UART_16550_COMPATIBLE == YES in coreConsultant. Value / Description: 0 MODEMSTAT 1 NOINTRPENDING 2 THREMPTY 4 RXDATAVAILABLE 6 RXLINESTAT 7 BUSYDETECT 12 CHARTIMEOUT	RO	0x1

fcr

FIFO Control Register

Module Instance	Base Address	Register Address
i_uart_0_uart	0xFFC02000	0xFFC02008
i_uart_1_uart	0xFFC02100	0xFFC02108

Offset: 0x8

Access: WO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								rt WO 0x0		tet WO 0x0		dmam WO 0x0	xfifor WO 0x0	rfifor WO 0x0	fifoe WO 0x0

fcr Fields

Bit	Name	Description	Access	Reset
7:6	rt	Bits[7:6], RCVR Trigger (or RT):. This is used to select the trigger level in the receiver FIFO at which the Received Data Available Interrupt will be generated. In auto flow control mode it is used to determine when the rts_n signal will be de-asserted. It also determines when the dma_rx_req_n signal will be asserted when in certain modes of operation. See section 5.9 on page 56 for details on DMA support. The following trigger levels are supported: 00 = 1 character in the FIFO 01 = FIFO 1/4 full 10 = FIFO 1/2 full 11 = FIFO 2 less than full **Value** **Description** 0 FIFO_CHAR_1 1 FIFO_QUARTER_FULL 2 FIFO_HALF_FULL 3 FIFO_FULL_2	WO	0x0

Bit	Name	Description	Access	Reset
5:4	tet	Bits[5:4], TX Empty Trigger (or TET): Writes will have no effect when THRE_MODE_USER == Disabled. This is used to select the empty threshold level at which the THRE Interrupts will be generated when the mode is active. It also determines when the dma_tx_req_n signal will be asserted when in certain modes of operation. See section 5.9 on page 56 for details on DMA support. The following trigger levels are supported: 00 = FIFO empty 01 = 2 characters in the FIFO 10 = FIFO 1/4 full 11 = FIFO full **Value** **Description** 0 FIFO_EMPTY 1 FIFO_CHAR_2 2 FIFO_QUARTER_FULL 3 FIFO_HALF_FULL	WO	0x0
3	dmam	Bit[3], DMA Mode (or DMAM): This determines the DMA signalling mode used for the dma_tx_req_n and dma_rx_req_n output signals when additional DMA handshaking signals are not selected (DMA_EXTRA == NO). See section 5.9 on page 56 for details on DMA support. 0 = mode 0 1 = mode 1 **Value** **Description** 0 MODE0 1 MODE1	WO	0x0

Bit	Name	Description	Access	Reset
2	xfifor	Bit[2], XMIT FIFO Reset (or XFIFOR): This resets the control portion of the transmit FIFO and treats the FIFO as empty. This will also de-assert the DMA TX request and single signals when additional DMA handshaking signals are selected (DMA_EXTRA == YES). Note that this bit is 'self-clearing' and it is not necessary to clear this bit. **Value** **Description** 1 RESET	WO	0x0
1	rfifor	Bit[1], RCVR FIFO Reset (or RFIFOR): This resets the control portion of the receive FIFO and treats the FIFO as empty. This will also de-assert the DMA RX request and single signals when additional DMA handshaking signals are selected (DMA_EXTRA == YES). Note that this bit is 'self-clearing' and it is not necessary to clear this bit. **Value** **Description** 1 RESET	WO	0x0
0	fifoe	Bit[0], FIFO Enable (or FIFOE): This enables/disables the transmit (XMIT) and receive (RCVR) FIFO's. Whenever the value of this bit is changed both the XMIT and RCVR controller portion of FIFO's will be reset. **Value** **Description** 0 DISABLED 1 ENABLED	WO	0x0

lcr

Line Control Register

Module Instance	Base Address	Register Address
i_uart_0_uart	0xFFC02000	0xFFC0200C
i_uart_1_uart	0xFFC02100	0xFFC0210C

Offset: 0xC

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
rsvd_lcr_31to8 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rsvd_lcr_31to8 RO 0x0								dlab RW 0x0	break RW 0x0	sp RW 0x0	eps RW 0x0	pen RW 0x0	stop RW 0x0	dls RW 0x0	

lcr Fields

Bit	Name	Description	Access	Reset
31:8	rsvd_lcr_31to8	Reserved bits [31:8] - Read Only	RO	0x0
7	dlab	Divisor Latch Access Bit. If UART_16550_COMPATIBLE == NO then, writeable only when UART is not busy (USR[0] is zero), otherwise always writable, always readable. This bit is used to enable reading and writing of the Divisor Latch register (DLL and DLH) to set the baud rate of the UART. This bit must be cleared after initial baud rate setup in order to access other registers.	RW	0x0

Bit	Name	Description	Access	Reset
6	break	Break Control Bit. This is used to cause a break condition to be transmitted to the receiving device. If set to one the serial output is forced to the spacing (logic 0) state. When not in Loopback Mode, as determined by MCR[4], the sout line is forced low until the Break bit is cleared. If SIR_MODE == Enabled and active (MCR[6] set to one) the sir_out_n line is continuously pulsed. When in Loopback Mode, the break condition is internally looped back to the receiver and the sir_out_n line is forced low.	RW	0x0

5	sp	From DW_apb_uart_regfile.sv:
		// aaraujo @ 17/05/2011 : CRM_9000431453
		// Stick parity lcr_ir[5] is now programmable
		lcr_ir[5:0] <= ipwdata[5:0];
		Value **Description**
		0 DISABLED
		1 ENABLED
			RW	0x0

Bit	Name	Description	Access	Reset
4	eps	Even Parity Select. If UART_16550_COMPATIBLE == NO then, writeable only when UART is not busy (USR[0] is zero), otherwise always writable, always readable. This is used to select between even and odd parity, when parity is enabled (PEN set to one). If set to one, an even number of logic '1's is transmitted or checked. If set to zero, an odd number of logic '1's is transmitted or checked. **Value** **Description** 0 ODDPAR 1 EVENPAR	RW	0x0
3	pen	Parity Enable. If UART_16550_COMPATIBLE == NO then, writeable only when UART is not busy (USR[0] is zero), otherwise always writable, always readable. This bit is used to enable and disable parity generation and detection in transmitted and received serial character respectively. 0 = parity disabled 1 = parity enabled **Value** **Description** 0 DISABLED 1 ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
2	stop	Number of stop bits. If UART_16550_COMPATIBLE == NO then, writeable only when UART is not busy (USR[0] is zero), otherwise always writable, always readable. This is used to select the number of stop bits per character that the peripheral will transmit and receive. If set to zero, one stop bit is transmitted in the serial data. If set to one and the data bits are set to 5 (LCR[1:0] set to zero) one and a half stop bits is transmitted. Otherwise, two stop bits are transmitted. Note that regardless of the number of stop bits selected the receiver will only check the first stop bit. 0 = 1 stop bit 1 = 1.5 stop bits when DLS (LCR[1:0]) is zero, else 2 stop bit **Value** **Description** 0 ONESTOP 1 ONEPOINT5STOP	RW	0x0

Bit	Name	Description	Access	Reset
1:0	dls	Data Length Select. If UART_16550_COMPATIBLE == NO then, writeable only when UART is not busy (USR[0] is zero), otherwise always writable, always readable. This is used to select the number of data bits per character that the peripheral will transmit and receive. The number of bit that may be selected areas follows: 00 = 5 bits 01 = 6 bits 10 = 7 bits 11 = 8 bits **Value** **Description** 0 LENGTH5 1 LENGTH6 2 LENGTH7 3 LENGTH8	RW	0x0

mcr

Modem Control Register

Module Instance	Base Address	Register Address
i_uart_0_uart	0xFFC02000	0xFFC02010
i_uart_1_uart	0xFFC02100	0xFFC02110

Offset: 0x10

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
rsvd_mcr_31to7 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rsvd_mcr_31to7 RO 0x0									sire RO 0x0	afce RW 0x0	loopback RW 0x0	out2 RW 0x0	out1 RW 0x0	rts RW 0x0	dtr RW 0x0

mcr Fields

Bit	Name	Description	Access	Reset
31:7	rsvd_mcr_31to7	Reserved bits [31:7] - Read Only	RO	0x0
6	sire	SIR Mode Enable. Writeable only when SIR_MODE == Enabled, always readable. This is used to enable/ disable the IrDA SIR Mode features as described in section 5.2 on page 47. 0 = IrDA SIR Mode disabled 1 = IrDA SIR Mode enabled	RO	0x0
5	afce	Auto Flow Control Enable. Writeable only when AFCE_MODE == Enabled, always readable. When FIFOs are enabled and the Auto Flow Control Enable (AFCE) bit is set, Auto Flow Control features are enabled as described in section 5.6 on page 51. 0 = Auto Flow Control Mode disabled 1 = Auto Flow Control Mode enabled **Value** **Description** 0 DISABLED 1 ENABLED	RW	0x0

Bit	Name	Description	Access	Reset
4	loopback	LoopBack Bit. This is used to put the UART into a dDW_iagnostic mode for test purposes. If operating in UART mode (SIR_MODE != Enabled OR NOT active, MCR[6] set to zero), data on the sout line is held high, while serial data output is looped back to the sin line, internally. In this mode all the interrupts are fully functional. Also, in loopback mode, the modem control inputs (dsr_n, cts_n, ri_n, dcd_n) are disconnected and the modem control outputs (dtr_n, rts_n, out1_n, out2_n) are looped back to the inputs, internally. If operating in infrared mode (SIR_MODE == Enabled AND active, MCR[6] set to one), data on the sir_out_n line is held low, while serial data output is inverted and looped back to the sir_in line.	RW	0x0
3	out2	OUT2. This is used to directly control the user-designated Output2 (out2_n) output. The value written to this location is inverted and driven out on out2_n, that is: 0 = out2_n de-asserted (logic 1) 1 = out2_n asserted (logic 0) Note that in Loopback mode (MCR[4] set to one), the out2_n output is held inactive high while the value of this location is internally looped back to an input. **Value** **Description** 0 LOGIC1 1 LOGIC0	RW	0x0

Bit	Name	Description	Access	Reset
2	out1	OUT1. This is used to directly control the user-designated Output1 (out1_n) output. The value written to this location is inverted and driven out on out1_n, that is: 0 = out1_n de-asserted (logic 1) 1 = out1_n asserted (logic 0) Note that in Loopback mode (MCR[4] set to one), the out1_n output is held inactive high while the value of this location is internally looped back to an input. **Value** **Description** 0 LOGIC1 1 LOGIC0	RW	0x0

Bit	Name	Description	Access	Reset
1	rts	Request to Send. This is used to directly control the Request to Send (rts_n) output. The Request To Send (rts_n) output is used to inform the modem or data set that the UART is ready to exchange data. When Auto RTS Flow Control is not enabled (MCR[5] set to zero), the rts_n signal is set low by programming MCR[1] (RTS) to a high. In Auto Flow Control, AFCE_MODE == Enabled and active (MCR[5] set to one) and FIFO's enable (FCR[0] set to one), the rts_n output is controlled in the same way, but is also gated with the receiver FIFO threshold trigger (rts_n is inactive high when above the threshold). The rts_n signal will be de-asserted when MCR[1] is set low. Note that in Loopback mode (MCR[4] set to one), the rts_n output is held inactive high while the value of this location is internally looped back to an input. **Value** **Description** 0 LOGIC1 1 LOGIC0	RW	0x0

Bit	Name	Description	Access	Reset
0	dtr	Data Terminal Ready. This is used to directly control the Data Terminal Ready (dtr_n) output. The value written to this location is inverted and driven out on dtr_n, that is: 0 = dtr_n de-asserted (logic 1) 1 = dtr_n asserted (logic 0) The Data Terminal Ready output is used to inform the modem or data set that the UART is ready to establish communi-cations. Note that in Loopback mode (MCR[4] set to one), the dtr_n output is held inactive high while the value of this location is internally looped back to an input. **Value** **Description** 0 LOGIC1 1 LOGIC0	RW	0x0

lsr

Line Status Register

Module Instance	Base Address	Register Address
i_uart_0_uart	0xFFC02000	0xFFC02014
i_uart_1_uart	0xFFC02100	0xFFC02114

Offset: 0x14

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
rsvd_lsr_31to8 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rsvd_lsr_31to8 RO 0x0								rfe RO 0x0	temt RO 0x1	thre RO 0x1	bi RO 0x0	fe RO 0x0	pe RO 0x0	oe RO 0x0	dr RO 0x0

lsr Fields

Bit	Name	Description	Access	Reset
31:8	rsvd_lsr_31to8	Reserved bits [31:8] - Read Only	RO	0x0
7	rfe	Receiver FIFO Error bit. This bit is only relevant when FIFO_MODE != NONE AND FIFO's are enabled (FCR[0] set to one). This is used to indicate if there is at least one parity error, framing error, or break indication in the FIFO. That is: 0 = no error in RX FIFO 1 = error in RX FIFO This bit is cleared when the LSR is read and the character with the error is at the top of the receiver FIFO and there are no subsequent errors in the FIFO. **Value** **Description** 0 NOERR 1 ERR	RO	0x0

Bit	Name	Description	Access	Reset
6	temt	Transmitter Empty bit. If in FIFO mode (FIFO_MODE != NONE) and FIFO's enabled (FCR[0] set to one), this bit is set whenever the Transmitter Shift Register and the FIFO are both empty. If in the non-FIFO mode or FIFO's are disabled, this bit is set whenever the Transmitter Holding Register and the Transmitter Shift Register are both empty. **Value** **Description** 0 NOTEMPTY 1 EMPTY	RO	0x1
5	thre	Transmit Holding Register Empty bit. If THRE_MODE_USER == Disabled or THRE mode is disabled (IER[7] set to zero) and regardless of FIFO's being implemented/enabled or not, this bit indicates that the THR or TX FIFO is empty. This bit is set whenever data is transferred from the THR or TX FIFO to the transmitter shift register and no new data has been written to the THR or TX FIFO. This also causes a THRE Interrupt to occur, if the THRE Interrupt is enabled. If THRE_MODE_USER == Enabled AND FIFO_MODE != NONE and both modes are active (IER[7] set to one and FCR[0] set to one respectively), the function-ality is switched to indicate the transmitter FIFO is full, and no longer controls THRE interrupts, which are then controlled by the FCR[5:4] threshold setting. Program-mable THRE interrupt mode operation is described in detail in section 5.7 on page 52.	RO	0x1

Bit	Name	Description	Access	Reset
4	bi	Break Interrupt bit. This is used to indicate the detection of a break sequence on the serial input data. If in UART mode it is set whenever the serial input, sin, is held in a logic '0' state for longer than the sum of start time + data bits + parity + stop bits. If in infrared mode it is set whenever the serial input, sir_in, is continuously pulsed to logic '0' for longer than the sum of start time + data bits + parity + stop bits. A break condition on serial input causes one and only one character, consisting of all zeros, to be received by the UART. In the FIFO mode, the character associated with the break condition is carried through the FIFO and is revealed when the character is at the top of the FIFO. Reading the LSR clears the BI bit. In the non-FIFO mode, the BI indication occurs immediately and persists until the LSR is read.	RO	0x0

Bit	Name	Description	Access	Reset
3	fe	Framing Error bit. This is used to indicate the occurrence of a framing error in the receiver. A framing error occurs when the receiver does not detect a valid STOP bit in the received data. In the FIFO mode, since the framing error is associated with a character received, it is revealed when the character with the framing error is at the top of the FIFO. When a framing error occurs the UART will try resynchronize. It does this by assuming that the error was due to the start bit of the next character and then continues receiving the other bit i.e. data, and/or parity and stop. It should be noted that the Framing Error (FE) bit (LSR[3]) will be set if a break interrupt has occurred, as indicated by Break Interrupt (BI) bit (LSR[4]). 0 = no framing error 1 = framing error Reading the LSR clears the FE bit. **Value** **Description** 0 NOFRAMEERR 1 FRAMEERR	RO	0x0

Bit	Name	Description	Access	Reset
2	pe	Parity Error bit. This is used to indicate the occurrence of a parity error in the receiver if the Parity Enable (PEN) bit (LCR[3]) is set. In the FIFO mode, since the parity error is associated with a character received, it is revealed when the character with the parity error arrives at the top of the FIFO. It should be noted that the Parity Error (PE) bit (LSR[2]) will be set if a break interrupt has occurred, as indicated by Break Interrupt (BI) bit (LSR[4]). 0 = no parity error 1 = parity error Reading the LSR clears the PE bit. **Value** **Description** 0 NOPARITYERR 1 PARITYERR	RO	0x0

Bit	Name	Description	Access	Reset
1	oe	Overrun error bit. This is used to indicate the occurrence of an overrun error. This occurs if a new data character was received before the previous data was read. In the non-FIFO mode, the OE bit is set when a new character arrives in the receiver before the previous character was read from the RBR. When this happens, the data in the RBR is overwritten. In the FIFO mode, an overrun error occurs when the FIFO is full and a new character arrives at the receiver. The data in the FIFO is retained and the data in the receive shift register is lost. 0 = no overrun error 1 = overrun error Reading the LSR clears the OE bit. **Value** / **Description** 0 — NOOVERRUN 1 — OVERRUN	RO	0x0
0	dr	Data Ready bit. This is used to indicate that the receiver contains at least one character in the RBR or the receiver FIFO. 0 = no data ready 1 = data ready This bit is cleared when the RBR is read in the non-FIFO mode, or when the receiver FIFO is empty, in the FIFO mode. **Value** / **Description** 0 — NODATARDY 1 — DATARDY	RO	0x0

msr

Modem Status Register

```
It should be noted that whenever bits 0, 1, 2 or 3 is set to logic
one, to indicate
a change on the modem control inputs, a modem status interrupt will
be generated
if enabled via the IER regardless of when the change occurred. Since
the delta bits
(bits 0, 1, 3) can get set after a reset if their respective modem
signals are
active (see individual bits for details), a read of the MSR after
reset can be
performed to prevent unwanted interrupts.
```

Module Instance	Base Address	Register Address
i_uart_0_uart	0xFFC02000	0xFFC02018
i_uart_1_uart	0xFFC02100	0xFFC02118

Offset: 0x18

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
rsvd_msc_31to8 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rsvd_msc_31to8 RO 0x0								dcd RO 0x0	ri RO 0x0	dsr RO 0x0	cts RO 0x0	ddcd RO 0x0	teri RO 0x0	ddsr RO 0x0	dcts RO 0x0

msr Fields

Bit	Name	Description	Access	Reset
31:8	rsvd_msc_31to8	Reserved bits [31:8] - Read Only	RO	0x0

Bit	Name	Description	Access	Reset
7	dcd	Data Carrier Detect. This is used to indicate the current state of the modem control line dcd_n. That is this bit is the complement dcd_n. When the Data Carrier Detect input (dcd_n) is asserted it is an indication that the carrier has been detected by the modem or data set. 0 = dcd_n input is de-asserted (logic 1) 1 = dcd_n input is asserted (logic 0) In Loopback Mode (MCR[4] set to one), DCD is the same as MCR[3] (Out2). **Value** **Description** 0 LOGIC1 1 LOGIC0	RO	0x0
6	ri	Ring Indicator. This is used to indicate the current state of the modem control line ri_n. That is this bit is the complement ri_n. When the Ring Indicator input (ri_n) is asserted it is an indication that a telephone ringing signal has been received by the modem or data set. 0 = ri_n input is de-asserted (logic 1) 1 = ri_n input is asserted (logic 0) In Loopback Mode (MCR[4] set to one), RI is the same as MCR[2] (Out1). **Value** **Description** 0 LOGIC1 1 LOGIC0	RO	0x0

Bit	Name	Description	Access	Reset
5	dsr	Data Set Ready. This is used to indicate the current state of the modem control line dsr_n. That is this bit is the complement dsr_n. When the Data Set Ready input (dsr_n) is asserted it is an indication that the modem or data set is ready to establish communications with the DW_apb_uart. 0 = dsr_n input is de-asserted (logic 1) 1 = dsr_n input is asserted (logic 0) In Loopback Mode (MCR[4] set to one), DSR is the same as MCR[0] (DTR). **Value** **Description** 0 LOGIC1 1 LOGIC0	RO	0x0
4	cts	Clear to Send. This is used to indicate the current state of the modem control line cts_n. That is, this bit is the complement cts_n. When the Clear to Send input (cts_n) is asserted it is an indication that the modem or data set is ready to exchange data with the DW_apb_uart. 0 = cts_n input is de-asserted (logic 1) 1 = cts_n input is asserted (logic 0) In Loopback Mode (MCR[4] set to one), CTS is the same as MCR[1] (RTS). **Value** **Description** 0 LOGIC1 1 LOGIC0	RO	0x0

Bit	Name	Description	Access	Reset
3	ddcd	Delta Data Carrier Detect. This is used to indicate that the modem control line dcd_n has changed since the last time the MSR was read. That is: 0 = no change on dcd_n since last read of MSR 1 = change on dcd_n since last read of MSR Reading the MSR clears the DDCD bit. In Loopback Mode (MCR[4] set to one), DDCD reflects changes on MCR[3] (Out2). Note, if the DDCD bit is not set and the dcd_n signal is asserted (low) and a reset occurs (software or otherwise), then the DDCD bit will get set when the reset is removed if the dcd_n signal remains asserted. **Value** **Description** 0 NOCHG 1 CHG	RO	0x0
2	teri	Trailing Edge of Ring Indicator. This is used to indicate that a change on the input ri_n (from an active low, to an inactive high state) has occurred since the last time the MSR was read. That is: 0 = no change on ri_n since last read of MSR 1 = change on ri_n since last read of MSR Reading the MSR clears the TERI bit. In Loopback Mode (MCR[4] set to one), TERI reflects when MCR[2] (Out1) has changed state from a high to a low. **Value** **Description** 0 NOCHG 1 CHG	RO	0x0

Bit	Name	Description	Access	Reset
1	ddsr	Delta Data Set Ready. This is used to indicate that the modem control line dsr_n has changed since the last time the MSR was read. That is: 0 = no change on dsr_n since last read of MSR 1 = change on dsr_n since last read of MSR Reading the MSR clears the DDSR bit. In Loopback Mode (MCR[4] set to one), DDSR reflects changes on MCR[0] (DTR). Note, if the DDSR bit is not set and the dsr_n signal is asserted (low) and a reset occurs (software or otherwise), then the DDSR bit will get set when the reset is removed if the dsr_n signal remains asserted. **Value** **Description** 0 NOCHG 1 CHG	RO	0x0

Bit	Name	Description	Access	Reset
0	dcts	Delta Clear to Send. This is used to indicate that the modem control line cts_n has changed since the last time the MSR was read. That is: 0 = no change on cts_n since last read of MSR 1 = change on cts_n since last read of MSR Reading the MSR clears the DCTS bit. In Loopback Mode (MCR[4] set to one), DCTS reflects changes on MCR[1] (RTS). Note, if the DCTS bit is not set and the cts_n signal is asserted (low) and a reset occurs (software or otherwise), then the DCTS bit will get set when the reset is removed if the cts_n signal remains asserted. **Value** **Description** 0 NOCHG 1 CHG	RO	0x0

scr

Scratchpad Register

Module Instance	Base Address	Register Address
i_uart_0_uart	0xFFC02000	0xFFC0201C
i_uart_1_uart	0xFFC02100	0xFFC0211C

Offset: 0x1C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
rsvd_scr_31to8 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rsvd_scr_31to8 RO 0x0								scr RW 0x0							

scr Fields

Bit	Name	Description	Access	Reset
31:8	rsvd_scr_31to8	Reserved bits [31:8] - Read Only	RO	0x0
7:0	scr	This register is for programmers to use as a temporary storage space. It has no defined purpose in the DW_apb_uart.	RW	0x0

srbr_sthr_0

This is multi-function register. It is shadow register for Receive Buffer Register and Transmit Holding Register.

Module Instance	Base Address	Register Address
i_uart_0_uart	0xFFC02000	0xFFC02030
i_uart_1_uart	0xFFC02100	0xFFC02130

Offset: 0x30

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
rsvd_srbr_sthr_0_31to8 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rsvd_srbr_sthr_0_31to8 RO 0x0								srbr_sthr_0 RO 0x0							

srbr_sthr_0 Fields

Bit	Name	Description	Access	Reset
31:8	rsvd_srbr_sthr_0_31to8	Reserved bits [31:8] - Read Only	RO	0x0

Bit	Name	Description	Access	Reset
7:0	srbr_sthr_0	This is shadow register for RBR and THR and has been allocated sixteen 32-bit locations so as to accomodate burst accesses from the master. srbr : This register contains the data byte received on the serial input port (sin) in UART mode or the serial infrared input (sir_in) in infrared mode. The data in this register is valid only if the Data Ready (DR) bit in the Line status Register (LSR) is set. If in non-FIFO mode (FIFO_MODE == NONE) or FIFOs are disabled (FCR[0] set to zero), the data in the RBR must be read before the next data arrives, otherwise it is overwritten, resulting in an overrun error. If in FIFO mode (FIFO_MODE != NONE) and FIFOs are enabled (FCR[0] set to one), this register accesses the head of the receive FIFO. If the receive FIFO is full and this register is not read before the next data character arrives, then the data already in the FIFO are preserved, but any incoming data is lost. An overrun error also occurs. sthr: This register contains data to be transmitted on the serial output port (sout) in UART mode or the serial infrared output (sir_out_n) in infrared mode. Data should only be written to the THR when the THR Empty (THRE) bit (LSR[5]) is set. If in non-FIFO mode or FIFOs are disabled (FCR[0] set to zero) and THRE is set, writing a single character to the THR clears the THRE. Any additional writes to the THR before the THRE is set again causes the THR data to be overwritten. If in FIFO mode and FIFOs are enabled (FCR[0] set to one) and THRE is set, x number of characters of data may be written to the THR before the FIFO is full. The number x (default=16) is determined by the value of FIFO Depth that you set during configura-tion. Any attempt to write data when the FIFO is full results in	RO	0x0

srbr_sthr_1

This is multi-function register. It is shadow register for Receive Buffer Register and Transmit Holding Register.

Module Instance	Base Address	Register Address
i_uart_0_uart	0xFFC02000	0xFFC02034
i_uart_1_uart	0xFFC02100	0xFFC02134

Offset: 0x34

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
rsvd_srbr_sthr_1_31to8 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rsvd_srbr_sthr_1_31to8 RO 0x0								srbr_sthr_1 RO 0x0							

srbr_sthr_1 Fields

Bit	Name	Description	Access	Reset
31:8	rsvd_srbr_sthr_1_31to8	Reserved bits [31:8] - Read Only	RO	0x0
7:0	srbr_sthr_1	See srbr_sthr_0 description	RO	0x0

srbr_sthr_2

This is multi-function register. It is shadow register for Receive Buffer Register and Transmit Holding Register.

Module Instance	Base Address	Register Address
i_uart_0_uart	0xFFC02000	0xFFC02038

Module Instance	Base Address	Register Address
i_uart_1_uart	0xFFC02100	0xFFC02138

Offset: 0x38

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
rsvd_srbr_sthr_2_31to8 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rsvd_srbr_sthr_2_31to8 RO 0x0								srbr_sthr_2 RO 0x0							

srbr_sthr_2 Fields

Bit	Name	Description	Access	Reset
31:8	rsvd_srbr_sthr_2_31to8	Reserved bits [31:8] - Read Only	RO	0x0
7:0	srbr_sthr_2	See srbr_sthr_0 description	RO	0x0

srbr_sthr_3

This is multi-function register. It is shadow register for Receive Buffer Register and Transmit Holding Register.

Module Instance	Base Address	Register Address
i_uart_0_uart	0xFFC02000	0xFFC0203C
i_uart_1_uart	0xFFC02100	0xFFC0213C

Offset: 0x3C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
rsvd_srbr_sthr_3_31to8 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rsvd_srbr_sthr_3_31to8 RO 0x0								srbr_sthr_3 RO 0x0							

srbr_sthr_3 Fields

Bit	Name	Description	Access	Reset
31:8	rsvd_srbr_sthr_3_31to8	Reserved bits [31:8] - Read Only	RO	0x0
7:0	srbr_sthr_3	See srbr_sthr_0 description	RO	0x0

srbr_sthr_4

This is multi-function register. It is shadow register for Receive Buffer Register and Transmit Holding Register.

Module Instance	Base Address	Register Address
i_uart_0_uart	0xFFC02000	0xFFC02040
i_uart_1_uart	0xFFC02100	0xFFC02140

Offset: 0x40

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
rsvd_srbr_sthr_4_31to8 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rsvd_srbr_sthr_4_31to8 RO 0x0								srbr_sthr_4 RO 0x0							

srbr_sthr_4 Fields

Bit	Name	Description	Access	Reset
31:8	rsvd_srbr_sthr_4_31to8	Reserved bits [31:8] - Read Only	RO	0x0
7:0	srbr_sthr_4	See srbr_sthr_0 description	RO	0x0

srbr_sthr_5

This is multi-function register. It is shadow register for Receive Buffer Register and Transmit Holding Register.

Module Instance	Base Address	Register Address
i_uart_0_uart	0xFFC02000	0xFFC02044
i_uart_1_uart	0xFFC02100	0xFFC02144

Offset: 0x44

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
rsvd_srbr_sthr_5_31to8 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rsvd_srbr_sthr_5_31to8 RO 0x0								srbr_sthr_5 RO 0x0							

srbr_sthr_5 Fields

Bit	Name	Description	Access	Reset
31:8	rsvd_srbr_sthr_5_31to8	Reserved bits [31:8] - Read Only	RO	0x0
7:0	srbr_sthr_5	See srbr_sthr_0 description	RO	0x0

srbr_sthr_6

This is multi-function register. It is shadow register for Receive Buffer Register and Transmit Holding Register.

Module Instance	Base Address	Register Address
i_uart_0_uart	0xFFC02000	0xFFC02048
i_uart_1_uart	0xFFC02100	0xFFC02148

Offset: 0x48

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
rsvd_srbr_sthr_6_31to8 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rsvd_srbr_sthr_6_31to8 RO 0x0								srbr_sthr_6 RO 0x0							

srbr_sthr_6 Fields

Bit	Name	Description	Access	Reset
31:8	rsvd_srbr_sthr_6_31to8	Reserved bits [31:8] - Read Only	RO	0x0
7:0	srbr_sthr_6	See srbr_sthr_0 description	RO	0x0

srbr_sthr_7

This is multi-function register. It is shadow register for Receive Buffer Register and Transmit Holding Register.

Module Instance	Base Address	Register Address
i_uart_0_uart	0xFFC02000	0xFFC0204C

Module Instance	Base Address	Register Address
i_uart_1_uart	0xFFC02100	0xFFC0214C

Offset: 0x4C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
rsvd_srbr_sthr_7_31to8 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rsvd_srbr_sthr_7_31to8 RO 0x0								srbr_sthr_7 RO 0x0							

srbr_sthr_7 Fields

Bit	Name	Description	Access	Reset
31:8	rsvd_srbr_sthr_7_31to8	Reserved bits [31:8] - Read Only	RO	0x0
7:0	srbr_sthr_7	See srbr_sthr_0 description	RO	0x0

srbr_sthr_8

This is multi-function register. It is shadow register for Receive Buffer Register and Transmit Holding Register.

Module Instance	Base Address	Register Address
i_uart_0_uart	0xFFC02000	0xFFC02050
i_uart_1_uart	0xFFC02100	0xFFC02150

Offset: 0x50

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
rsvd_srbr_sthr_8_31to8 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rsvd_srbr_sthr_8_31to8 RO 0x0								srbr_sthr_8 RO 0x0							

srbr_sthr_8 Fields

Bit	Name	Description	Access	Reset
31:8	rsvd_srbr_sthr_8_31to8	Reserved bits [31:8] - Read Only	RO	0x0
7:0	srbr_sthr_8	See srbr_sthr_0 description	RO	0x0

srbr_sthr_9

This is multi-function register. It is shadow register for Receive Buffer Register and Transmit Holding Register.

Module Instance	Base Address	Register Address
i_uart_0_uart	0xFFC02000	0xFFC02054
i_uart_1_uart	0xFFC02100	0xFFC02154

Offset: 0x54

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
rsvd_srbr_sthr_9_31to8 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rsvd_srbr_sthr_9_31to8 RO 0x0								srbr_sthr_9 RO 0x0							

srbr_sthr_9 Fields

Bit	Name	Description	Access	Reset
31:8	rsvd_srbr_sthr_9_31to8	Reserved bits [31:8] - Read Only	RO	0x0
7:0	srbr_sthr_9	See srbr_sthr_0 description	RO	0x0

srbr_sthr_10

This is multi-function register. It is shadow register for Receive Buffer Register and Transmit Holding Register.

Module Instance	Base Address	Register Address
i_uart_0_uart	0xFFC02000	0xFFC02058
i_uart_1_uart	0xFFC02100	0xFFC02158

Offset: 0x58

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
rsvd_srbr_sthr_10_31to8 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rsvd_srbr_sthr_10_31to8 RO 0x0								srbr_sthr_10 RO 0x0							

srbr_sthr_10 Fields

Bit	Name	Description	Access	Reset
31:8	rsvd_srbr_sthr_10_31to8	Reserved bits [31:8] - Read Only	RO	0x0
7:0	srbr_sthr_10	See srbr_sthr_0 description	RO	0x0

srbr_sthr_11

This is multi-function register. It is shadow register for Receive Buffer Register and Transmit Holding Register.

Module Instance	Base Address	Register Address
i_uart_0_uart	0xFFC02000	0xFFC0205C
i_uart_1_uart	0xFFC02100	0xFFC0215C

Offset: 0x5C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
rsvd_srbr_sthr_11_31to8 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rsvd_srbr_sthr_11_31to8 RO 0x0								srbr_sthr_11 RO 0x0							

srbr_sthr_11 Fields

Bit	Name	Description	Access	Reset
31:8	rsvd_srbr_sthr_11_31to8	Reserved bits [31:8] - Read Only	RO	0x0
7:0	srbr_sthr_11	See srbr_sthr_0 description	RO	0x0

srbr_sthr_12

This is multi-function register. It is shadow register for Receive Buffer Register and Transmit Holding Register.

Module Instance	Base Address	Register Address
i_uart_0_uart	0xFFC02000	0xFFC02060

Module Instance	Base Address	Register Address
i_uart_1_uart	0xFFC02100	0xFFC02160

Offset: 0x60

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
rsvd_srbr_sthr_12_31to8 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rsvd_srbr_sthr_12_31to8 RO 0x0								srbr_sthr_12 RO 0x0							

srbr_sthr_12 Fields

Bit	Name	Description	Access	Reset
31:8	rsvd_srbr_sthr_12_31to8	Reserved bits [31:8] - Read Only	RO	0x0
7:0	srbr_sthr_12	See srbr_sthr_0 description	RO	0x0

srbr_sthr_13

This is multi-function register. It is shadow register for Receive Buffer Register and Transmit Holding Register.

Module Instance	Base Address	Register Address
i_uart_0_uart	0xFFC02000	0xFFC02064
i_uart_1_uart	0xFFC02100	0xFFC02164

Offset: 0x64

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
rsvd_srbr_sthr_13_31to8 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rsvd_srbr_sthr_13_31to8 RO 0x0								srbr_sthr_13 RO 0x0							

srbr_sthr_13 Fields

Bit	Name	Description	Access	Reset
31:8	rsvd_srbr_sthr_13_31to8	Reserved bits [31:8] - Read Only	RO	0x0
7:0	srbr_sthr_13	See srbr_sthr_0 description	RO	0x0

srbr_sthr_14

This is multi-function register. It is shadow register for Receive Buffer Register and Transmit Holding Register.

Module Instance	Base Address	Register Address
i_uart_0_uart	0xFFC02000	0xFFC02068
i_uart_1_uart	0xFFC02100	0xFFC02168

Offset: 0x68

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
rsvd_srbr_sthr_14_31to8 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rsvd_srbr_sthr_14_31to8 RO 0x0								srbr_sthr_14 RO 0x0							

srbr_sthr_14 Fields

Bit	Name	Description	Access	Reset
31:8	rsvd_srbr_sthr_14_31to8	Reserved bits [31:8] - Read Only	RO	0x0
7:0	srbr_sthr_14	See srbr_sthr_0 description	RO	0x0

srbr_sthr_15

This is multi-function register. It is shadow register for Receive Buffer Register and Transmit Holding Register.

Module Instance	Base Address	Register Address
i_uart_0_uart	0xFFC02000	0xFFC0206C
i_uart_1_uart	0xFFC02100	0xFFC0216C

Offset: 0x6C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
rsvd_srbr_sthr_15_31to8 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rsvd_srbr_sthr_15_31to8 RO 0x0								srbr_sthr_15 RO 0x0							

srbr_sthr_15 Fields

Bit	Name	Description	Access	Reset
31:8	rsvd_srbr_sthr_15_31to8	Reserved bits [31:8] - Read Only	RO	0x0
7:0	srbr_sthr_15	See srbr_sthr_0 description	RO	0x0

far

FIFO Access Register

Module Instance	Base Address	Register Address
i_uart_0_uart	0xFFC02000	0xFFC02070
i_uart_1_uart	0xFFC02100	0xFFC02170

Offset: 0x70

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
rsvd_far_31to1 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rsvd_far_31to1 RO 0x0															srbr_sthr RW 0x0

far Fields

Bit	Name	Description	Access	Reset
31:1	rsvd_far_31to1	Reserved bits [31:1] - Read Only	RO	0x0

Bit	Name	Description	Access	Reset
0	srbr_sthr	Writes will have no effect when FIFO_ACCESS == No, always readable. This register is use to enable a FIFO access mode for testing, so that the receive FIFO can be written by the master and the transmit FIFO can be read by the master when FIFO's are implemented and enabled. When FIFO's are not implemented or not enabled it allows the RBR to be written by the master and the THR to be read by the master. 0 = FIFO access mode disabled 1 = FIFO access mode enabled Note, that when the FIFO access mode is enabled/disabled, the control portion of the receive FIFO and transmit FIFO is reset and the FIFO's are treated as empty. **Value** **Description** 0 DISABLED 1 ENABLED	RW	0x0

tfr

Transmit FIFO Read

Module Instance	Base Address	Register Address
i_uart_0_uart	0xFFC02000	0xFFC02074
i_uart_1_uart	0xFFC02100	0xFFC02174

Offset: 0x74

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
rsvd_tfr_31to8 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rsvd_tfr_31to8 RO 0x0								tfr RO 0x0							

tfr Fields

Bit	Name	Description	Access	Reset
31:8	rsvd_tfr_31to8	Reserved bits [31:8] - Read Only	RO	0x0
7:0	tfr	Transmit FIFO Read. These bits are only valid when FIFO access mode is enabled (FAR[0] is set to one). When FIFO's are implemented and enabled, reading this register gives the data at the top of the transmit FIFO. Each consecutive read pops the transmit FIFO and gives the next data value that is currently at the top of the FIFO. When FIFO's are not implemented or not enabled, reading this register gives the data in the THR.	RO	0x0

rfw

Receive FIFO Write

Module Instance	Base Address	Register Address
i_uart_0_uart	0xFFC02000	0xFFC02078
i_uart_1_uart	0xFFC02100	0xFFC02178

Offset: 0x78

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
rsvd_rfw_31to10 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rsvd_rfw_31to10 RO 0x0						rffe WO 0x0	rfpe WO 0x0	rfwd WO 0x0							

rfw Fields

Bit	Name	Description	Access	Reset
31:10	rsvd_rfw_31to10	Reserved bits [31:10] - Read Only	RO	0x0
9	rffe	Receive FIFO Framing Error. These bits are only valid when FIFO access mode is enabled (FAR[0] is set to one). When FIFO's are implemented and enabled, this bit is used to write framing error detection information to the receive FIFO. When FIFO's are not implemented or not enabled, this bit is used to write framing error detection information to the RBR.	WO	0x0
8	rfpe	Receive FIFO Parity Error. These bits are only valid when FIFO access mode is enabled (FAR[0] is set to one). When FIFO's are implemented and enabled, this bit is used to write parity error detection information to the receive FIFO. When FIFO's are not implemented or not enabled, this bit is used to write parity error detection information to the RBR.	WO	0x0

Bit	Name	Description	Access	Reset
7:0	rfwd	Receive FIFO Write Data. These bits are only valid when FIFO access mode is enabled (FAR[0] is set to one). When FIFO's are implemented and enabled, the data that is written to the RFWD is pushed into the receive FIFO. Each consecutive write pushes the new data to the next write location in the receive FIFO. When FIFO's are not implemented or not enabled, the data that is written to the RFWD is pushed into the RBR.	WO	0x0

usr

UART Status register.

Module Instance	Base Address	Register Address
i_uart_0_uart	0xFFC02000	0xFFC0207C
i_uart_1_uart	0xFFC02100	0xFFC0217C

Offset: 0x7C

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
rsvd_usr_31to5 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rsvd_usr_31to5 RO 0x0											rff RO 0x0	rfne RO 0x0	tfe RO 0x1	tfnf RO 0x1	rsvd_busy RO 0x0

usr Fields

Bit	Name	Description	Access	Reset
31:5	rsvd_usr_31to5	Reserved bits [31:5] - Read Only	RO	0x0
4	rff	Receive FIFO Full. This bit is only valid when FIFO_STAT == YES. This is used to indicate that the receive FIFO is completely full. That is: 0 = Receive FIFO not full 1 = Receive FIFO Full This bit is cleared when the RX FIFO is no longer full. **Value** — **Description** 0 — NOTFULL 1 — FULL	RO	0x0
3	rfne	Receive FIFO Not Empty. This bit is only valid when FIFO_STAT == YES. This is used to indicate that the receive FIFO contains one or more entries. 0 = Receive FIFO is empty 1 = Receive FIFO is not empty This bit is cleared when the RX FIFO is empty. **Value** — **Description** 0 — EMPTY 1 — NOTEMPTY	RO	0x0

Bit	Name	Description	Access	Reset
2	tfe	Transmit FIFO Empty. This bit is only valid when FIFO_STAT == YES. This is used to indicate that the transmit FIFO is completely empty. 0 = Transmit FIFO is not empty 1 = Transmit FIFO is empty This bit is cleared when the TX FIFO is no longer empty. **Value** **Description** 0 NOTEMPTY 1 EMPTY	RO	0x1
1	tfnf	Transmit FIFO Not Full. This bit is only valid when FIFO_STAT == YES. This is used to indicate that the transmit FIFO in not full. 0 = Transmit FIFO is full 1 = Transmit FIFO is not full This bit is cleared when the TX FIFO is full. **Value** **Description** 0 FULL 1 NOTFULL	RO	0x1

Bit	Name	Description	Access	Reset
0	rsvd_busy	UART Busy. This bit is only valid when UART_16550_COMPATIBLE == NO. This indicates that a serial transfer is in progress, when cleared indicates that the DW_apb_uart is idle or inactive. 0 = DW_apb_uart is idle or inactive 1 - DW_apb_uart is busy (actively transferring data) Note that it is possible for the UART Busy bit to be cleared even though a new character may have been sent from another device. That is, if the DW_apb_uart has no data in the THR and RBR and there is no transmission in progress and a start bit of a new character has just reached the DW_apb_uart. This is due to the fact that a valid start is not seen until the middle of the bit period and this duration is dependent on the baud divisor that has been programmed. If a second system clock has been implemented (CLOCK_MODE == Enabled) the assertion of this bit will also be delayed by several cycles of the slower clock.	RO	0x0

tfl

Module Instance	Base Address	Register Address
i_uart_0_uart	0xFFC02000	0xFFC02080
i_uart_1_uart	0xFFC02100	0xFFC02180

Offset: 0x80

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
rsvd_tfl_31toaddr_width RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rsvd_tfl_31toaddr_width RO 0x0								tfl RO 0x0							

tfl Fields

Bit	Name	Description	Access	Reset
31:8	rsvd_tfl_31toaddr_width	Reserved bits: 31 downto addr bus width + 1 - Read Only	RO	0x0
7:0	tfl	Transmit FIFO Level. This is indicates the number of data entries in the transmit FIFO.	RO	0x0

rfl

Receive FIFO Level.

Module Instance	Base Address	Register Address
i_uart_0_uart	0xFFC02000	0xFFC02084
i_uart_1_uart	0xFFC02100	0xFFC02184

Offset: 0x84

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
rsvd_rfl_31toaddr_width RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rsvd_rfl_31toaddr_width RO 0x0								rfl RO 0x0							

rfl Fields

Bit	Name	Description	Access	Reset
31:8	rsvd_rfl_31toaddr_width	Reserved bits: 31 downnto addr bus width + 1 - Read Only	RO	0x0
7:0	rfl	Receive FIFO Level. This is indicates the number of data entries in the receive FIFO.	RO	0x0

srr

Software Reset Register.

Module Instance	Base Address	Register Address
i_uart_0_uart	0xFFC02000	0xFFC02088
i_uart_1_uart	0xFFC02100	0xFFC02188

Offset: 0x88

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
rsvd_srr_31to3 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rsvd_srr_31to3 RO 0x0													xfr WO 0x0	rfr WO 0x0	ur WO 0x0

srr Fields

Bit	Name	Description	Access	Reset
31:3	rsvd_srr_31to3	Reserved bits [31:3] - Read Only	RO	0x0
2	xfr	XMIT FIFO Reset. Writes will have no effect when FIFO_MODE == NONE. This is a shadow register for the XMIT FIFO Reset bit (FCR[2]). This can be used to remove the burden on software having to store previously written FCR values (which are pretty static) just to reset the transmit FIFO. This resets the control portion of the transmit FIFO and treats the FIFO as empty. This will also de-assert the DMA TX request and single signals when additional DMA handshaking signals are selected **Value** **Description** 0 NORESET 1 RESET	WO	0x0

Bit	Name	Description	Access	Reset
1	rfr	RCVR FIFO Reset. Writes will have no effect when FIFO_MODE == NONE. This is a shadow register for the RCVR FIFO Reset bit (FCR[1]). This can be used to remove the burden on software having to store previously written FCR values (which are pretty static) just to reset the reeive FIFO. This resets the control portion of the receive FIFO and treats the FIFO as empty. This will also de-assert the DMA RX request and single signals when additional DMA handshaking signals are selected (DMA_EXTRA == YES). Note that this bit is 'self-clearing' and it is not necessary to clear this bit. **Value** **Description** 0 NORESET 1 RESET	WO	0x0
0	ur	UART Reset. This asynchronously resets the DW_apb_uart and synchronously removes the reset assertion. For a two clock implementation both pclk and sclk domains will be reset. **Value** **Description** 0 NORESET 1 RESET	WO	0x0

srts

Shadow Request to Send.

Module Instance	Base Address	Register Address
i_uart_0_uart	0xFFC02000	0xFFC0208C

Module Instance	Base Address	Register Address
i_uart_1_uart	0xFFC02100	0xFFC0218C

Offset: 0x8C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
rsvd_srts_31to1 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rsvd_srts_31to1 RO 0x0															srts RW 0x0

srts Fields

Bit	Name	Description	Access	Reset
31:1	rsvd_srts_31to1	Reserved bits [31:1] - Read Only	RO	0x0

Bit	Name	Description	Access	Reset
0	srts	Shadow Request to Send. This is a shadow register for the RTS bit (MCR[1]), this can be used to remove the burden of having to performing a read modify write on the MCR. This is used to directly control the Request to Send (rts_n) output. The Request To Send (rts_n) output is used to inform the modem or data set that the UART is ready to exchange data. When Auto RTS Flow Control is not enabled (MCR[5] set to zero), the rts_n signal is set low by programming MCR[1] (RTS) to a high. In Auto Flow Control, AFCE_MODE == Enabled and active (MCR[5] set to one) and FIFO's enable (FCR[0] set to one), the rts_n output is controlled in the same way, but is also gated with the receiver FIFO threshold trigger (rts_n is inactive high when above the threshold) only when RTC Flow Trigger is disabled; otherwise it is gated by the receiver FIFO almost-full trigger, where almost full refers to two available slots in the FIFO (rts_n is inactive high when above the threshold). Note that in Loopback mode (MCR[4] set to one), the rts_n output is held inactive high while the value of this location is internally looped back to an input. **Value** **Description** 0 LOGIC1 1 LOGIC0	RW	0x0

sbcr

Shadow Break Control Register.

Module Instance	Base Address	Register Address
i_uart_0_uart	0xFFC02000	0xFFC02090
i_uart_1_uart	0xFFC02100	0xFFC02190

Offset: 0x90

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
rsvd_sbcr_31to1 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rsvd_sbcr_31to1 RO 0x0															sbcr RW 0x0

sbcr Fields

Bit	Name	Description	Access	Reset
31:1	rsvd_sbcr_31to1	Reserved bits [31:1] - Read Only	RO	0x0

Bit	Name	Description	Access	Reset
0	sbcr	Shadow Break Control Bit. This is a shadow register for the Break bit (LCR[6]), this can be used to remove the burden of having to performing a read modify write on the LCR. This is used to cause a break condition to be transmitted to the receiving device. If set to one the serial output is forced to the spacing (logic 0) state. When not in Loopback Mode, as determined by MCR[4], the sout line is forced low until the Break bit is cleared. If SIR_MODE == Enabled and active (MCR[6] set to one) the sir_out_n line is continuously pulsed. When in Loopback Mode, the break condition is internally looped back to the receiver. Value: 0 — DISABLED; 1 — ENABLED	RW	0x0

sdmam

Shadow DMA Mode.

Module Instance	Base Address	Register Address
i_uart_0_uart	0xFFC02000	0xFFC02094
i_uart_1_uart	0xFFC02100	0xFFC02194

Offset: 0x94

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
rsvd_sdmam_31to1 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rsvd_sdmam_31to1 RO 0x0															sdmam RW 0x0

sdmam Fields

Bit	Name	Description	Access	Reset
31:1	rsvd_sdmam_31to1	Reserved bits [31:1] - Read Only	RO	0x0
0	sdmam	Shadow DMA Mode. This is a shadow register for the DMA mode bit (FCR[3]). This can be used to remove the burden of having to store the previously written value to the FCR in memory and having to mask this value so that only the DMA Mode bit gets updated. This determines the DMA signalling mode used for the dma_tx_req_n and dma_rx_req_n output signals when additional DMA handshaking signals are not selected (DMA_EXTRA == NO). See section 5.9 on page 54 for details on DMA support. 0 = mode 0 1 = mode 1 **Value** **Description** 0 SINGLE 1 MULTIPLE	RW	0x0

sfe

Shadow FIFO Enable

Module Instance	Base Address	Register Address
i_uart_0_uart	0xFFC02000	0xFFC02098

Module Instance	Base Address	Register Address
i_uart_1_uart	0xFFC02100	0xFFC02198

Offset: 0x98

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
rsvd_sfe_31to1 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rsvd_sfe_31to1 RO 0x0															sfe RW 0x0

sfe Fields

Bit	Name	Description	Access	Reset
31:1	rsvd_sfe_31to1	Reserved bits [31:1] - Read Only	RO	0x0
0	sfe	Shadow FIFO Enable. This is a shadow register for the FIFO enable bit (FCR[0]). This can be used to remove the burden of having to store the previously written value to the FCR in memory and having to mask this value so that only the FIFO enable bit gets updated. This enables/disables the transmit (XMIT) and receive (RCVR) FIFO's. If this bit is set to zero (disabled) after being enabled then both the XMIT and RCVR controller portion of FIFO's will be reset. **Value** — **Description** 0 — DISABLED 1 — ENABLED	RW	0x0

srt

Shadow RCVR Trigger

Module Instance	Base Address	Register Address
i_uart_0_uart	0xFFC02000	0xFFC0209C
i_uart_1_uart	0xFFC02100	0xFFC0219C

Offset: 0x9C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
rsvd_srt_31to2 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rsvd_srt_31to2 RO 0x0														srt RW 0x0	

srt Fields

Bit	Name	Description	Access	Reset
31:2	rsvd_srt_31to2	Reserved bits [31:2] - Read Only	RO	0x0

Bit	Name	Description	Access	Reset
1:0	srt	Shadow RCVR Trigger. This is a shadow register for the RCVR trigger bits (FCR[7:6]). This can be used to remove the burden of having to store the previously written value to the FCR in memory and having to mask this value so that only the RCVR trigger bit gets updated. This is used to select the trigger level in the receiver FIFO at which the Received Data Available Interrupt will be generated. It also determines when the dma_rx_req_n signal will be asserted when DMA Mode (FCR[3]) is set to one. The following trigger levels are supported: 00 = 1 character in the FIFO 01 = FIFO ■full 10 = FIFO ■full 11 = FIFO 2 less than full **Value** **Description** 0 ONECHAR 1 QUARTERFULL 2 HALFFULL 3 FULLLESS2	RW	0x0

stet

Shadow TX Empty Trigger

Module Instance	Base Address	Register Address
i_uart_0_uart	0xFFC02000	0xFFC020A0
i_uart_1_uart	0xFFC02100	0xFFC021A0

Offset: 0xA0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
rsvd_stet_31to2 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rsvd_stet_31to2 RO 0x0														stet RW 0x0	

stet Fields

Bit	Name	Description	Access	Reset
31:2	rsvd_stet_31to2	Reserved bits [31:2] - Read Only	RO	0x0

Bit	Name	Description	Access	Reset
1:0	stet	Shadow TX Empty Trigger. This is a shadow register for the TX empty trigger bits (FCR[5:4]). This can be used to remove the burden of having to store the previously written value to the FCR in memory and having to mask this value so that only the TX empty trigger bit gets updated. Writes will have no effect when THRE_MODE_USER == Disabled. This is used to select the empty threshold level at which the THRE Interrupts will be generated when the mode is active. These threshold levels are also described in. The following trigger levels are supported: 00 = FIFO empty 01 = 2 characters in the FIFO 10 = FIFO ■full 11 = FIFO ■full **Value** **Description** 0 FIFOEMPTY 1 TWOCHARS 2 QUARTERFULL 3 HALFFULL	RW	0x0

htx

Halt TX

Module Instance	Base Address	Register Address
i_uart_0_uart	0xFFC02000	0xFFC020A4
i_uart_1_uart	0xFFC02100	0xFFC021A4

Offset: 0xA4

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
rsvd_htx_31to1 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rsvd_htx_31to1 RO 0x0															htx RW 0x0

htx Fields

Bit	Name	Description	Access	Reset
31:1	rsvd_htx_31to1	Reserved bits [31:1] - Read Only	RO	0x0
0	htx	Halt TX. Writes will have no effect when FIFO_MODE == NONE, always readable. This register is use to halt transmissions for testing, so that the transmit FIFO can be filled by the master when FIFO's are implemented and enabled. Note, if FIFO's are implemented and not enabled the setting of the halt TX register will have no effect on operation. 0 = Halt TX disabled 1 = Halt TX enabled **Value** **Description** 0 DISABLED 1 ENABLED	RW	0x0

dmasa

DMA Software Acknowledge

Module Instance	Base Address	Register Address
i_uart_0_uart	0xFFC02000	0xFFC020A8

Module Instance	Base Address	Register Address
i_uart_1_uart	0xFFC02100	0xFFC021A8

Offset: 0xA8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
rsvd_dmasa_31to1 RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rsvd_dmasa_31to1 RO 0x0															dmasa WO 0x0

dmasa Fields

Bit	Name	Description	Access	Reset
31:1	rsvd_dmasa_31to1	Reserved bits [31:1] - Read Only	RO	0x0
0	dmasa	DMA Software Acknowledge. Writes will have no effect when DMA_ EXTRA == No. This register is use to perform DMA software acknowledge if a transfer needs to be terminated due to an error condition. For example, if the DMA disables the channel, then the DW_ apb_uart should clear its request. This will cause the TX request, TX single, RX request and RX single signals to de-assert. Note that this bit is 'self- clearing' and it is not necessary to clear this bit.	WO	0x0

cpr

Component Parameter Register

Module Instance	Base Address	Register Address
i_uart_0_uart	0xFFC02000	0xFFC020F4
i_uart_1_uart	0xFFC02100	0xFFC021F4

Offset: 0xF4

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
rsvd_cpr_31to24 RO 0x0								fifo_mode RO 0x8							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rsvd_cpr_15to14 RO 0x0		dma_extra RO 0x1	uart_add_encoded_param RO 0x1	shadow RO 0x1	fifo_stat RO 0x1	fifo_access RO 0x1	additional_feat RO 0x1	sir_lp_mode RO 0x0	sir_mode RO 0x0	thre_mode RO 0x1	afce_mode RO 0x1	rsvd_cpr_3to2 RO 0x0		apbdatawidth RO 0x2	

cpr Fields

Bit	Name	Description	Access	Reset
31:24	rsvd_cpr_31to24	Reserved bits [31:24] - Read Only	RO	0x0
23:16	fifo_mode	Encoding of FIFO_MODE configuration parameter value.DW_apb_uart.ralf 0x00 = 0, 0x01 = 16, 0x02 = 32, toset 0x80 = 2048, 0x81- 0xff = reserved **Value** **Description** 128 FIFO128BYTES	RO	0x8
15:14	rsvd_cpr_15to14	Reserved bits [15:14] - Read Only	RO	0x0

Bit	Name	Description	Access	Reset
13	dma_extra	Encoding of DMA_EXTRA configuration parameter value. 0 = FALSE,DW_apb_uart.ralf 1 = TRUE **Value** **Description** 1 ENABLED	RO	0x1
12	uart_add_encoded_param	Encoding of UART_ADD_ENCODED_PARAMS configuration parameter value. 0 = FALSE, 1 = TRUE **Value** **Description** 1 ENABLED	RO	0x1
11	shadow	Encoding of SHADOW configuration parameter value. 0 = FALSE, 1 = TRUE **Value** **Description** 1 ENABLED	RO	0x1
10	fifo_stat	Encoding of FIFO_STAT configuration parameter value. 0 = FALSE, 1 = TRUE **Value** **Description** 1 ENABLED	RO	0x1
9	fifo_access	Encoding of FIFO_ACCESS configuration parameter value. 0 = FALSE, 1 = TRUE **Value** **Description** 1 ENABLED	RO	0x1

Bit	Name	Description	Access	Reset
8	additional_feat	Encoding of ADDITIONAL_FEATURES configuration parameter value. 0 = FALSE, 1 = TRUE **Value** **Description** 1 ENABLED	RO	0x1
7	sir_lp_mode	Encoding of SIR_LP_MODE configuration parameter value. 0 = FALSE, 1 = TRUE **Value** **Description** 0 DISABLED	RO	0x0
6	sir_mode	Encoding of SIR_MODE configuration parameter value. 0 = FALSE, 1 = TRUE **Value** **Description** 0 DISABLED	RO	0x0
5	thre_mode	Encoding of THRE_MODE configuration parameter value. 0 = FALSE, 1 = TRUE **Value** **Description** 1 ENABLED	RO	0x1
4	afce_mode	Encoding of AFCE_MODE configuration parameter value. 0 = FALSE, 1 = TRUE **Value** **Description** 1 ENABLED	RO	0x1
3:2	rsvd_cpr_3to2	Reserved bits [3:2] - Read Only	RO	0x0

Bit	Name	Description	Access	Reset
1:0	apbdatawidth	Encoding of APB_DATA_WIDTH configuration parameter value. 00 = 8 bits, 01 = 16 bits, 10 = 32 bits, 11 = reserved **Value** **Description** 2 WIDTH32BITS	RO	0x2

ucv

Component Version

Module Instance	Base Address	Register Address
i_uart_0_uart	0xFFC02000	0xFFC020F8
i_uart_1_uart	0xFFC02100	0xFFC021F8

Offset: 0xF8

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
uart_component_version RO 0x3331342A															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
uart_component_version RO 0x3331342A															

ucv Fields

Bit	Name	Description	Access	Reset
31:0	uart_component_version	ASCII value for each number in the version, followed by *. For example 32_30_31_2A represents the version 2.01*	RO	0x3331342A

ctr

Component Type Register

Module Instance	Base Address	Register Address
i_uart_0_uart	0xFFC02000	0xFFC020FC
i_uart_1_uart	0xFFC02100	0xFFC021FC

Offset: 0xFC

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
peripheral_id RO 0x44570110															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
peripheral_id RO 0x44570110															

ctr Fields

Bit	Name	Description	Access	Reset
31:0	peripheral_id	This register contains the peripherals identification code.	RO	0x44570110

Document Revision History

Table 21-5: Document Revision History

Date	Version	Changes
October 2016	2016.10.28	Maintenance release.
May 2016	2016.05.03	Maintenance release.
November 2015	2015.11.02	Renamed *Interface Pins* section to *HPS I/O Pins* and moved this section and *FPGA Routing* under *UART Controller Signal Description*
May 2015	2015.05.04	Maintenance release.
December 2014	2014.12.15	• Maintenance release. • Added *Taking the UART Out of Reset* section.

Date	Version	Changes
August 2014	2014.08.18	Initial release.

General-Purpose I/O Interface 22

2016.10.28

a10_5v4 Subscribe Send Feedback

The hard processor system (HPS) provides three general-purpose I/O (GPIO) interface modules. The GPIO modules are instances of the Synopsys® DesignWare® APB General Purpose Programming I/O (DW_apb_gpio) peripheral.† (55)

Features of the GPIO Interface

The GPIO interface offers the following features:

- Supports digital debounce
- Configurable interrupt mode
- Supports up to 62 I/O pins

(55) Portions © 2016 Synopsys, Inc. Used with permission. All rights reserved. Synopsys & DesignWare are registered trademarks of Synopsys, Inc. All documentation is provided "as is" and without any warranty. Synopsys expressly disclaims any and all warranties, express, implied, or otherwise, including the implied warranties of merchantability, fitness for a particular purpose, and non-infringement, and any warranties arising out of a course of dealing or usage of trade.

†Paragraphs marked with the dagger (†) symbol are Synopsys Proprietary. Used with permission.

GPIO Interface Block Diagram and System Integration

The figure below shows a block diagram of the GPIO interface. The following table shows a pin table of the GPIO interface:

Figure 22-1: Arria 10 SoC GPIO

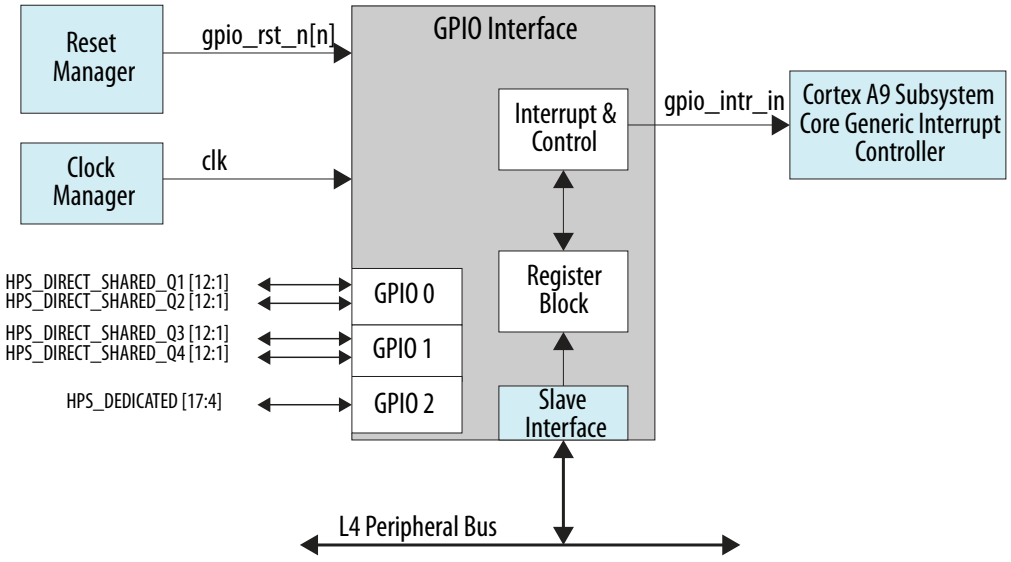


Table 22-1: GPIO Interface pin table

Pin Name	Mapped to GPIO Signal Name	Comments
HPS_DIRECT_SHARED_Q1 [12:1]	GPIO 0 [11:0]	Shared Input / Output
HPS_DIRECT_SHARED_Q2 [12:1]	GPIO 0 [23:12]	Shared Input / Output
HPS_DIRECT_SHARED_Q3 [12:1]	GPIO 1 [11:0]	Shared Input / Output
HPS_DIRECT_SHARED_Q4 [12:1]	GPIO 1 [23:12]	Shared Input / Output
HPS_DEDICATED [17:4]	GPIO 2 [13:0]	Dedicated Input / Output

Table 22-2: GPIO Interface pin table

Pin Name	Mapped to GPIO Signal Name	Comments
HPS_DEDICATED_Q1 [12:1]	GPIO 0 [11:0]	Input / Output
HPS_DEDICATED_Q2 [12:1]	GPIO 0 [23:12]	Input / Output
HPS_DEDICATED_Q3 [12:1]	GPIO 1 [11:0]	Input / Output

Pin Name	Mapped to GPIO Signal Name	Comments
HPS_DEDICATED_Q4 [12:1]	GPIO 1 [23:12]	Input / Output

Functional Description of the GPIO Interface

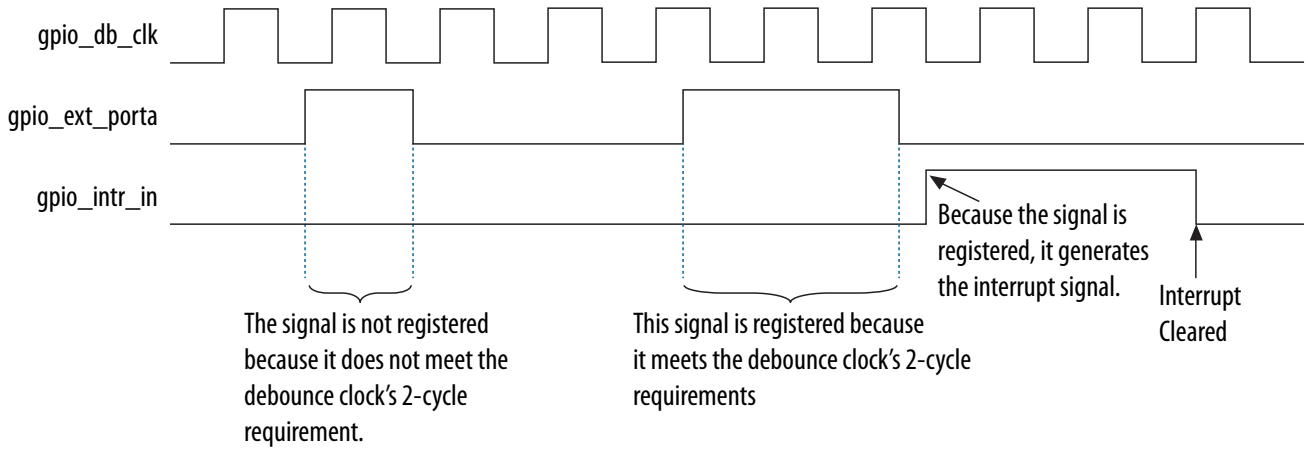
Debounce Operation

The GPIO modules provided in the HPS include optional debounce capabilities. The external signal can be debounced to remove any spurious glitches that are less than one period of the external debouncing clock, `gpio_db_clk`. †

When input signals are debounced using the `gpio_db_clk` debounce clock, the signals must be active for a minimum of two cycles of the debounce clock to guarantee that they are registered. Any input pulse widths less than a debounce clock period are filtered out. If the input signal pulse width is between one and two debounce clock widths, it may or may not be filtered out, depending on its phase relationship to the debounce clock. If the input pulse spans two rising edges of the debounce clock, it is registered. If it spans only one rising edge, it is not registered. †

The figure below shows a timing diagram of the debounce circuitry for both cases: a bounced input signal, and later, a propagated input signal.

Figure 22-2: Debounce Timing With Asychronous Reset Flip-Flops



Note: Enabling the debounce circuitry increases interrupt latency by two clock cycles of the debounce clock.

Pin Directions

All GPIO pins can be configured to be either input or output signals.

Taking the GPIO Interface Out of Reset

When a cold or warm reset is issued in the HPS, the reset manager resets this module and holds it in reset until software releases it.

After the Cortex-A9 MPCore CPU boots, it can deassert the reset signal by clearing the appropriate bits in the reset manager's corresponding reset register. For details about reset registers, refer to "Module Reset Signals".

GPIO Interface Programming Model

Debounce capability for each of the input signals can be enabled or disabled under software control by setting the corresponding bits in the `gpio_debounce` register, accordingly. The debounce clock must be stable and operational before the debounce capability is enabled.

Under software control, the direction of the external I/O pad is controlled by a write to the `gpio_swportx_ddr` register. When configured as input mode, reading `gpio_ext_porta` would read the values on the signal of the external I/O pad. When configured as output mode, the data written to the `gpio_swporta_dr` register drives the output buffer of the I/O pad. The same pins are shared for both input and output modes, so they cannot be configured as input and output modes at the same time. †

General-Purpose I/O Interface Address Map and Register Definitions

For complete HPS address map and register definitions, refer to the **Arria 10 HPS Address Map and Register Definitions**.

gpio Address Map

Module Instance	Base Address	End Address
i_gpio_0_gpio	0xFFC02900	0xFFC029FF
i_gpio_1_gpio	0xFFC02A00	0xFFC02AFF
i_gpio_2_gpio	0xFFC02B00	0xFFCF9FFF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
gpio_swporta_dr on page 22-16	0x0	32	RW	0x0	Name: Port A data register set_attribute $reg -attr RALAttribute -sub NO_BIT_BASH_TEST -value 1 Size: 1-32 bits Address Offset: 0x00 Read/Write Access: Read/Write

Register	Offset	Width	Access	Reset Value	Description
gpio_swporta_ddr on page 22-17	0x4	32	RW	0x0	Name: Port A Data Direction Register Size: 1-32 bits Address Offset: 0x04 Read/Write Access: Read/Write
gpio_inten on page 22-18	0x30	32	RW	0x0	Name: Interrupt enable register Size: 1-32 bits Address Offset: 0x30 Read/Write Access: Read/Write
gpio_intmask on page 22-19	0x34	32	RW	0x0	Name: Interrupt mask register Size: 1-32 bits Address Offset: 0x34 Read/Write Access: Read/Write
gpio_inttype_level on page 22-21	0x38	32	RW	0x0	Name: Interrupt level Size: 1-32 bits Address Offset: 0x38 Read/Write Access: Read/Write
gpio_int_polarity on page 22-22	0x3C	32	RW	0x0	Name: Interrupt polarity Size: 1-32 bits Address Offset: 0x3c Read/Write Access: Read/Write
gpio_intstatus on page 22-23	0x40	32	RO	0x0	Name: Interrupt status Size: 1-32 bits Address Offset: 0x40 Read/Write Access: Read/Write
gpio_raw_intstatus on page 22-24	0x44	32	RO	0x0	Name: Raw interrupt status Size: 1-32 bits Address Offset: 0x44 Read/Write Access: Read/Write

Register	Offset	Width	Access	Reset Value	Description
gpio_debounce on page 22-25	0x48	32	RW	0x0	Name: Debounce enable Size: 1-32 bits Address Offset: 0x48 Read/Write Access: Read/Write
gpio_porta_eoi on page 22-26	0x4C	32	WO	0x0	Name: Port A clear interrupt register Size: 1-32 bits Address Offset: 0x4c Read/Write Access: Write
gpio_ext_porta on page 22-27	0x50	32	RO	0x0	Name: Port A external port register Size: 1-32 bits Address Offset: 0x50 Read/Write Access: Read
gpio_ls_sync on page 22-28	0x60	32	RW	0x0	Name: Synchronization level Size: 1 bit Address Offset: 0x60 Read/Write Access: Read/Write
gpio_id_code on page 22-29	0x64	32	RO	0x0	Name: GPIO ID code Size: 1-32 bits Address Offset: 0x64 Read/Write Access: Read
gpio_ver_id_code on page 22-30	0x6C	32	RO	0x3230392A	Name: GPIO Component Version Size: 32 bits Address Offset: 0x6c Read/Write Access: Read
gpio_config_reg2 on page 22-31	0x70	32	RO	0x39CF7	Name: GPIO Configura-tion Register 2 Size: 32 bits Address Offset: 0x70 Read/Write Access: Read
gpio_config_reg1 on page 22-33	0x74	32	RO	0x1FF0F2	Name: GPIO Configura-tion Register 1 Size: 32 bits Address Offset: 0x74 Read/Write Access: Read

gpio Summary

Module Instance	Base Address
i_gpio_0_gpio	0xFFC02900
i_gpio_1_gpio	0xFFC02A00
i_gpio_2_gpio	0xFFC02B00

Register Address Offset	Bit Fields															
i_gpio_0_gpio																
gpio_swporta_dr 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved								gpio_swporta_dr RW 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	gpio_swporta_dr RW 0x0															
gpio_swporta_ddr 0x4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved								gpio_swporta_ddr RW 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	gpio_swporta_ddr RW 0x0															
gpio_inten 0x30	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved								gpio_inten RW 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	gpio_inten RW 0x0															
gpio_intmask 0x34	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved								gpio_intmask RW 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	gpio_intmask RW 0x0															

Register Address Offset	Bit Fields															
gpio_inttype_level 0x38	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved								gpio_inttype_level RW 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	gpio_inttype_level RW 0x0															
gpio_int_polarity 0x3C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved								gpio_int_polarity RW 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	gpio_int_polarity RW 0x0															
gpio_intstatus 0x40	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved								gpio_intstatus RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	gpio_intstatus RO 0x0															
gpio_raw_intstatus 0x44	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved								gpio_raw_intstatus RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	gpio_raw_intstatus RO 0x0															
gpio_debounce 0x48	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved								gpio_debounce RW 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	gpio_debounce RW 0x0															
gpio_porta_eoi 0x4C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved								gpio_porta_eoi WO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	gpio_porta_eoi WO 0x0															

Register Address Offset	Bit Fields															
gpio_ext_porta 0x50	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved								gpio_ext_porta RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	gpio_ext_porta RO 0x0															
gpio_ls_sync 0x60	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															gpio_ls_sync RW 0x0
gpio_id_code 0x64	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	gpio_id_code RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	gpio_id_code RO 0x0															
gpio_ver_id_code 0x6C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	gpio_ver_id_code RO 0x3230392A															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	gpio_ver_id_code RO 0x3230392A															
gpio_config_reg2 0x70	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved												encoded_id_pwidth_d RO 0x7			
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	encoded_id_pwidth_d RO 0x7	encoded_id_pwidth_c RO 0x7					encoded_id_pwidth_b RO 0x7						encoded_id_pwidth_a RO 0x17			

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved											encoded_id_width RO 0x1F				
gpio_config_reg1 0x74	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	gpio_id RO 0x1	add_encoded_params RO 0x1	debounce RO 0x1	porta_intr RO 0x1	hw_portd RO 0x0	hw_portc RO 0x0	hw_portb RO 0x0	hw_porta RO 0x0	portd_single_ctl RO 0x1	portc_single_ctl RO 0x1	portb_single_ctl RO 0x1	porta_single_ctl RO 0x1	num_ports RO 0x0		apb_data_width RO 0x2	
i_gpio_1_gpio																
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gpio_swporta_dr 0x0	Reserved								gpio_swporta_dr RW 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	gpio_swporta_dr RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gpio_swporta_ddr 0x4	Reserved								gpio_swporta_ddr RW 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	gpio_swporta_ddr RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gpio_inten 0x30	Reserved								gpio_inten RW 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	gpio_inten RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gpio_intmask 0x34	Reserved								gpio_intmask RW 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	gpio_intmask RW 0x0															

Register Address Offset	Bit Fields															
gpio_inttype_level 0x38	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved								gpio_inttype_level RW 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	gpio_inttype_level RW 0x0															
gpio_int_polarity 0x3C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved								gpio_int_polarity RW 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	gpio_int_polarity RW 0x0															
gpio_intstatus 0x40	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved								gpio_intstatus RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	gpio_intstatus RO 0x0															
gpio_raw_intstatus 0x44	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved								gpio_raw_intstatus RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	gpio_raw_intstatus RO 0x0															
gpio_debounce 0x48	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved								gpio_debounce RW 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	gpio_debounce RW 0x0															
gpio_porta_eoi 0x4C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved								gpio_porta_eoi WO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	gpio_porta_eoi WO 0x0															

Register Address Offset	Bit Fields															
gpio_ext_porta 0x50	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved								gpio_ext_porta RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	gpio_ext_porta RO 0x0															
gpio_ls_sync 0x60	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															gpio_ls_sync RW 0x0
gpio_id_code 0x64	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	gpio_id_code RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	gpio_id_code RO 0x0															
gpio_ver_id_code 0x6C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	gpio_ver_id_code RO 0x3230392A															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	gpio_ver_id_code RO 0x3230392A															
gpio_config_reg2 0x70	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved												encoded_id_pwidth_d RO 0x7			
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	encoded_id_pwidth_d RO 0x7	encoded_id_pwidth_c RO 0x7					encoded_id_pwidth_b RO 0x7						encoded_id_pwidth_a RO 0x17			

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved											encoded_id_width RO 0x1F				
gpio_config_reg1 0x74	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	gpio_id RO 0x1	add_encoded_params RO 0x1	debounce RO 0x1	porta_intr RO 0x1	hw_portd RO 0x0	hw_portc RO 0x0	hw_portb RO 0x0	hw_porta RO 0x0	portd_single_ctl RO 0x1	portc_single_ctl RO 0x1	portb_single_ctl RO 0x1	porta_single_ctl RO 0x1	num_ports RO 0x0		apb_data_width RO 0x2	
i_gpio_2_gpio																
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved								gpio_swporta_dr RW 0x0							
gpio_swporta_dr 0x0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	gpio_swporta_dr RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved								gpio_swporta_ddr RW 0x0							
gpio_swporta_ddr 0x4	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	gpio_swporta_ddr RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved								gpio_inten RW 0x0							
gpio_inten 0x30	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	gpio_inten RW 0x0															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved								gpio_intmask RW 0x0							
gpio_intmask 0x34	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	gpio_intmask RW 0x0															

Register Address Offset	Bit Fields															
gpio_inttype_level 0x38	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved								gpio_inttype_level RW 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	gpio_inttype_level RW 0x0															
gpio_int_polarity 0x3C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved								gpio_int_polarity RW 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	gpio_int_polarity RW 0x0															
gpio_intstatus 0x40	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved								gpio_intstatus RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	gpio_intstatus RO 0x0															
gpio_raw_intstatus 0x44	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved								gpio_raw_intstatus RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	gpio_raw_intstatus RO 0x0															
gpio_debounce 0x48	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved								gpio_debounce RW 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	gpio_debounce RW 0x0															
gpio_porta_eoi 0x4C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved								gpio_porta_eoi WO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	gpio_porta_eoi WO 0x0															

Register Address Offset	Bit Fields															
gpio_ext_porta 0x50	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved								gpio_ext_porta RO 0x0							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	gpio_ext_porta RO 0x0															
gpio_ls_sync 0x60	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															gpio_ls_sync RW 0x0
gpio_id_code 0x64	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	gpio_id_code RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	gpio_id_code RO 0x0															
gpio_ver_id_code 0x6C	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	gpio_ver_id_code RO 0x3230392A															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	gpio_ver_id_code RO 0x3230392A															
gpio_config_reg2 0x70	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved												encoded_id_pwidth_d RO 0x7			
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	encoded_id_pwidth_d RO 0x7	encoded_id_pwidth_c RO 0x7					encoded_id_pwidth_b RO 0x7						encoded_id_pwidth_a RO 0x17			

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved											encoded_id_width RO 0x1F				
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
gpio_config_reg1 0x74	gpio_id RO 0x1	add_encoded_params RO 0x1	debounce RO 0x1	port a_intr RO 0x1	hw_port d RO 0x0	hw_port c RO 0x0	hw_port b RO 0x0	hw_port a RO 0x0	port d_single_ctl RO 0x1	port c_single_ctl RO 0x1	port b_single_ctl RO 0x1	port a_single_ctl RO 0x1	num_ports RO 0x0		apb_data_width RO 0x2	

gpio_swporta_dr

```
        Name: Port A data register
set_attribute $reg -attr RALAttribute -sub NO_BIT_BASH_TEST -value 1
Size: 1-32 bits
Address Offset: 0x00
Read/Write Access: Read/Write
```

Module Instance	Base Address	Register Address
i_gpio_0_gpio	0xFFC02900	0xFFC02900
i_gpio_1_gpio	0xFFC02A00	0xFFC02A00
i_gpio_2_gpio	0xFFC02B00	0xFFC02B00

Offset: 0x0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								gpio_swporta_dr RW 0x0							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
gpio_swporta_dr RW 0x0															

gpio_swporta_dr Fields

Bit	Name	Description	Access	Reset
23:0	gpio_swporta_dr	Values written to this register are output on the I/O signals for Port A if the corresponding data direction bits for Port A are set to Output mode and the corresponding control bit for Port A is set to Software mode. The value read back is equal to the last value written to this register.	RW	0x0

gpio_swporta_ddr

```
Name: Port A Data Direction Register
Size: 1-32 bits
Address Offset: 0x04
Read/Write Access: Read/Write
```

Module Instance	Base Address	Register Address
i_gpio_0_gpio	0xFFC02900	0xFFC02904
i_gpio_1_gpio	0xFFC02A00	0xFFC02A04
i_gpio_2_gpio	0xFFC02B00	0xFFC02B04

Offset: 0x4

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								gpio_swporta_ddr RW 0x0							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
gpio_swporta_ddr RW 0x0															

gpio_swporta_ddr Fields

Bit	Name	Description	Access	Reset
23:0	gpio_swporta_ddr	Values written to this register independently control the direction of the corresponding data bit in Port A. The default direction can be configured as input or output after system reset through the GPIO_DFLT_SRC_A parameter. 0 Input (default) 1 Output **Value** **Description** 0x0 IN 0x1 OUT	RW	0x0

gpio_inten

```
        Name: Interrupt enable register
Size: 1-32 bits
Address Offset: 0x30
Read/Write Access: Read/Write
```

Module Instance	Base Address	Register Address
i_gpio_0_gpio	0xFFC02900	0xFFC02930
i_gpio_1_gpio	0xFFC02A00	0xFFC02A30
i_gpio_2_gpio	0xFFC02B00	0xFFC02B30

Offset: 0x30

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								gpio_inten RW 0x0							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
gpio_inten RW 0x0															

gpio_inten Fields

Bit	Name	Description	Access	Reset
23:0	gpio_inten	Allows each bit of Port A to be configured for interrupts. By default the generation of interrupts is disabled. Whenever a 1 is written to a bit of this register, it configures the corresponding bit on Port A to become an interrupt; otherwise, Port A operates as a normal GPIO signal. Interrupts are disabled on the corresponding bits of Port A if the corresponding data direction register is set to Output or if Port A mode is set to Hardware. 0 Configure Port A bit as normal GPIO signal (default) 1 Configure Port A bit as interrupt **Value** **Description** 0x0 DISABLE 0x1 ENABLE	RW	0x0

gpio_intmask

```
        Name: Interrupt mask register
Size: 1-32 bits
Address Offset: 0x34
Read/Write Access: Read/Write
```

Module Instance	Base Address	Register Address
i_gpio_0_gpio	0xFFC02900	0xFFC02934

Module Instance	Base Address	Register Address
i_gpio_1_gpio	0xFFC02A00	0xFFC02A34
i_gpio_2_gpio	0xFFC02B00	0xFFC02B34

Offset: 0x34

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								gpio_intmask RW 0x0							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
gpio_intmask RW 0x0															

gpio_intmask Fields

Bit	Name	Description	Access	Reset
23:0	gpio_intmask	Controls whether an interrupt on Port A can create an interrupt for the interrupt controller by not masking it. By default, all interrupts bits are unmasked. Whenever a 1 is written to a bit in this register, it masks the interrupt generation capability for this signal; otherwise interrupts are allowed through. The unmasked status can be read as well as the resultant status after masking. 0 Interrupt bits are unmasked (default) 1 Mask interrupt **Value** **Description** 0x0 DISABLE 0x1 ENABLE	RW	0x0

gpio_inttype_level

```
        Name: Interrupt level
Size: 1-32 bits
Address Offset: 0x38
Read/Write Access: Read/Write
```

Module Instance	Base Address	Register Address
i_gpio_0_gpio	0xFFC02900	0xFFC02938
i_gpio_1_gpio	0xFFC02A00	0xFFC02A38
i_gpio_2_gpio	0xFFC02B00	0xFFC02B38

Offset: 0x38

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								gpio_inttype_level RW 0x0							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
gpio_inttype_level RW 0x0															

gpio_inttype_level Fields

Bit	Name	Description	Access	Reset
23:0	gpio_inttype_level	Controls the type of interrupt that can occur on Port A. Whenever a 0 is written to a bit of this register, it configures the interrupt type to be level-sensitive; otherwise, it is edge-sensitive. 0 Level-sensitive (default) 1 Edge-sensitive **Value** **Description** 0x0 LEVEL 0x1 EDGE	RW	0x0

gpio_int_polarity

```
        Name: Interrupt polarity
Size: 1-32 bits
Address Offset: 0x3c
Read/Write Access: Read/Write
```

Module Instance	Base Address	Register Address
i_gpio_0_gpio	0xFFC02900	0xFFC0293C
i_gpio_1_gpio	0xFFC02A00	0xFFC02A3C
i_gpio_2_gpio	0xFFC02B00	0xFFC02B3C

Offset: 0x3C

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								gpio_int_polarity RW 0x0							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
gpio_int_polarity RW 0x0															

gpio_int_polarity Fields

Bit	Name	Description	Access	Reset
23:0	gpio_int_polarity	Controls the polarity of edge or level sensitivity that can occur on input of Port A. Whenever a 0 is written to a bit of this register, it configures the interrupt type to falling-edge or active-low sensitive; otherwise, it is rising-edge or active-high sensitive. 0 Active-low (default) 1 Active-high **Value** **Description** 0x0 ACTLOW 0x1 ACTHIGH	RW	0x0

gpio_intstatus

```
        Name: Interrupt status
Size: 1-32 bits
Address Offset: 0x40
Read/Write Access: Read/Write
```

Module Instance	Base Address	Register Address
i_gpio_0_gpio	0xFFC02900	0xFFC02940
i_gpio_1_gpio	0xFFC02A00	0xFFC02A40
i_gpio_2_gpio	0xFFC02B00	0xFFC02B40

Offset: 0x40

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								gpio_intstatus RO 0x0							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
gpio_intstatus RO 0x0															

gpio_intstatus Fields

Bit	Name	Description	Access	Reset
23:0	gpio_intstatus	Interrupt status of Port A. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0

gpio_raw_intstatus

```
        Name: Raw interrupt status
Size: 1-32 bits
Address Offset: 0x44
Read/Write Access: Read/Write
```

Module Instance	Base Address	Register Address
i_gpio_0_gpio	0xFFC02900	0xFFC02944
i_gpio_1_gpio	0xFFC02A00	0xFFC02A44
i_gpio_2_gpio	0xFFC02B00	0xFFC02B44

Offset: 0x44

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								gpio_raw_intstatus RO 0x0							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
gpio_raw_intstatus RO 0x0															

gpio_raw_intstatus Fields

Bit	Name	Description	Access	Reset
23:0	gpio_raw_intstatus	Raw interrupt of status of Port A (premasking bits) **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0

gpio_debounce

```
        Name: Debounce enable
Size: 1-32 bits
Address Offset: 0x48
Read/Write Access: Read/Write
```

Module Instance	Base Address	Register Address
i_gpio_0_gpio	0xFFC02900	0xFFC02948
i_gpio_1_gpio	0xFFC02A00	0xFFC02A48
i_gpio_2_gpio	0xFFC02B00	0xFFC02B48

Offset: 0x48

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								gpio_debounce RW 0x0							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
gpio_debounce RW 0x0															

gpio_debounce Fields

Bit	Name	Description	Access	Reset
23:0	gpio_debounce	Controls whether an external signal that is the source of an interrupt needs to be debounced to remove any spurious glitches. Writing a 1 to a bit in this register enables the debouncing circuitry. A signal must be valid for two periods of an external clock before it is internally processed. 0 No debounce (default) 1 Enable debounce **Value** **Description** 0x0 DISABLE 0x1 ENABLE	RW	0x0

gpio_porta_eoi

Name: Port A clear interrupt register
Size: 1-32 bits
Address Offset: 0x4c
Read/Write Access: Write

Module Instance	Base Address	Register Address
i_gpio_0_gpio	0xFFC02900	0xFFC0294C
i_gpio_1_gpio	0xFFC02A00	0xFFC02A4C
i_gpio_2_gpio	0xFFC02B00	0xFFC02B4C

Offset: 0x4C

Access: WO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								gpio_porta_eoi WO 0x0							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
gpio_porta_eoi WO 0x0															

gpio_porta_eoi Fields

Bit	Name	Description	Access	Reset
23:0	gpio_porta_eoi	Controls the clearing of edge type interrupts from Port A. When a 1 is written into a corresponding bit of this register, the interrupt is cleared. All interrupts are cleared when Port A is not configured for interrupts. 0 No interrupt clear (default) 1 Clear interrupt **Value** **Description** 0x0 NOCLR 0x1 CLR	WO	0x0

gpio_ext_porta

```
        Name: Port A external port register
Size: 1-32 bits
Address Offset: 0x50
Read/Write Access: Read
```

Module Instance	Base Address	Register Address
i_gpio_0_gpio	0xFFC02900	0xFFC02950
i_gpio_1_gpio	0xFFC02A00	0xFFC02A50

Module Instance	Base Address	Register Address
i_gpio_2_gpio	0xFFC02B00	0xFFC02B50

Offset: 0x50

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								gpio_ext_porta RO 0x0							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
gpio_ext_porta RO 0x0															

gpio_ext_porta Fields

Bit	Name	Description	Access	Reset
23:0	gpio_ext_porta	When Port A is configured as Input, then reading this location reads the values on the signal. When the data direction of Port A is set as Output, reading this location reads the data register for Port A.	RO	0x0

gpio_ls_sync

```
        Name: Synchronization level
Size: 1 bit
Address Offset: 0x60
Read/Write Access: Read/Write
```

Module Instance	Base Address	Register Address
i_gpio_0_gpio	0xFFC02900	0xFFC02960
i_gpio_1_gpio	0xFFC02A00	0xFFC02A60
i_gpio_2_gpio	0xFFC02B00	0xFFC02B60

Offset: 0x60

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															gpio_ls_sync RW 0x0

gpio_ls_sync Fields

Bit	Name	Description	Access	Reset
0	gpio_ls_sync	Writing a 1 to this register results in all level-sensitive interrupts being synchronized to pclk_intr. 0 No synchronization to pclk_intr (default) 1 Synchronize to pclk_intr **Value** **Description** 0x0 NOSYNC 0x1 SYNC	RW	0x0

gpio_id_code

```
Name: GPIO ID code
Size: 1-32 bits
Address Offset: 0x64
Read/Write Access: Read
```

Module Instance	Base Address	Register Address
i_gpio_0_gpio	0xFFC02900	0xFFC02964
i_gpio_1_gpio	0xFFC02A00	0xFFC02A64
i_gpio_2_gpio	0xFFC02B00	0xFFC02B64

Offset: 0x64

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gpio_id_code RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
gpio_id_code RO 0x0															

gpio_id_code Fields

Bit	Name	Description	Access	Reset
31:0	gpio_id_code	This is a user-specified code that a system can read. It can be used for chip identification, and so on.	RO	0x0

gpio_ver_id_code

```
        Name: GPIO Component Version
Size: 32 bits
Address Offset: 0x6c
Read/Write Access: Read
```

Module Instance	Base Address	Register Address
i_gpio_0_gpio	0xFFC02900	0xFFC0296C
i_gpio_1_gpio	0xFFC02A00	0xFFC02A6C
i_gpio_2_gpio	0xFFC02B00	0xFFC02B6C

Offset: 0x6C

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
gpio_ver_id_code RO 0x3230392A															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
gpio_ver_id_code RO 0x3230392A															

gpio_ver_id_code Fields

Bit	Name	Description	Access	Reset
31:0	gpio_ver_id_code	ASCII value for each number in the version.	RO	0x3230392A

gpio_config_reg2

```
          Name: GPIO Configuration Register 2
Size: 32 bits
Address Offset: 0x70
Read/Write Access: Read
```

Module Instance	Base Address	Register Address
i_gpio_0_gpio	0xFFC02900	0xFFC02970
i_gpio_1_gpio	0xFFC02A00	0xFFC02A70
i_gpio_2_gpio	0xFFC02B00	0xFFC02B70

Offset: 0x70

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved												encoded_id_pwidth_d RO 0x7			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
encoded_id_pwidth_d RO 0x7	encoded_id_pwidth_c RO 0x7					encoded_id_pwidth_b RO 0x7						encoded_id_pwidth_a RO 0x17			

gpio_config_reg2 Fields

Bit	Name	Description	Access	Reset
19:15	encoded_id_pwidth_d	The value of this register is derived from the GPIO_PWIDTH_D configuration parameter. 0x0 = 8 bits 0x1 = 16 bits 0x2 = 32 bits 0x3 = Reserved **Value** **Description** 0x1c WIDTHLESSONE24BITS 0x7 WIDTHLESSONE8BITS	RO	0x7
14:10	encoded_id_pwidth_c	The value of this register is derived from the GPIO_PWIDTH_C configuration parameter. 0x0 = 8 bits 0x1 = 16 bits 0x2 = 32 bits 0x3 = Reserved **Value** **Description** 0x1c WIDTHLESSONE24BITS 0x7 WIDTHLESSONE8BITS	RO	0x7

Bit	Name	Description	Access	Reset
9:5	encoded_id_pwidth_b	The value of this register is derived from the GPIO_PWIDTH_B configuration parameter. 0x0 = 8 bits 0x1 = 16 bits 0x2 = 32 bits 0x3 = Reserved **Value** **Description** 0x1c WIDTHLESSONE24BITS 0x7 WIDTHLESSONE8BITS	RO	0x7
4:0	encoded_id_pwidth_a	The value of this register is derived from the GPIO_PWIDTH_A configuration parameter. 0x0 = 8 bits 0x1 = 16 bits 0x2 = 32 bits 0x3 = Reserved **Value** **Description** 0x1c WIDTHLESSONE24BITS 0x7 WIDTHLESSONE8BITS	RO	0x17

gpio_config_reg1

```
        Name: GPIO Configuration Register 1
Size: 32 bits
Address Offset: 0x74
Read/Write Access: Read
```

Module Instance	Base Address	Register Address
i_gpio_0_gpio	0xFFC02900	0xFFC02974
i_gpio_1_gpio	0xFFC02A00	0xFFC02A74
i_gpio_2_gpio	0xFFC02B00	0xFFC02B74

Offset: 0x74

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved											encoded_id_width RO 0x1F				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
gpio_id RO 0x1	add_encoded_params RO 0x1	debounce RO 0x1	porta_intr RO 0x1	hw_portd RO 0x0	hw_portc RO 0x0	hw_portb RO 0x0	hw_porta RO 0x0	portd_single_ctl RO 0x1	portc_single_ctl RO 0x1	portb_single_ctl RO 0x1	porta_single_ctl RO 0x1	num_ports RO 0x0		apb_data_width RO 0x2	

gpio_config_reg1 Fields

Bit	Name	Description	Access	Reset
20:16	encoded_id_width	The value of this register is derived from the GPIO_ID_WIDTH configuration parameter. **Value** **Description** 0x1f ENCIDWIDTH	RO	0x1F
15	gpio_id	The value of this register is derived from the GPIO_ID configuration parameter. 0 = Exclude 1 = Include **Value** **Description** 0x1 IDCODE	RO	0x1
14	add_encoded_params	The value of this register is derived from the GPIO_ADD_ENCODED_PARAMS configuration parameter. 0 = False 1 = True **Value** **Description** 0x1 ADDENCPARAMS	RO	0x1

Bit	Name	Description	Access	Reset
13	debounce	The value of this register is derived from the GPIO_DEBOUNCE configuration parameter. 0 = Exclude 1 = Include **Value** **Description** 0x1 DEBOUNCEA	RO	0x1
12	porta_intr	The value of this register is derived from the GPIO_PORTA_INTR configuration parameter. 0 = Exclude 1 = Include **Value** **Description** 0x1 PORTAINTERR	RO	0x1
11	hw_portd	The value of this register is derived from the GPIO_HW_PORTD configuration parameter. 0 = Exclude 1 = Include	RO	0x0
10	hw_portc	The value of this register is derived from the GPIO_HW_PORTC configuration parameter. 0 = Exclude 1 = Include	RO	0x0
9	hw_portb	The value of this register is derived from the GPIO_HW_PORTB configuration parameter. 0 = Exclude 1 = Include	RO	0x0

Bit	Name	Description	Access	Reset
8	hw_porta	The value of this register is derived from the GPIO_HW_PORTA configuration parameter. 0 = Exclude 1 = Include **Value** **Description** 0x0 PORTANOHARD	RO	0x0
7	portd_single_ctl	The value of this register is derived from the GPIO_PORTD_SINGLE_CTL configuration parameter. 0 = False 1 = True **Value** **Description** 0x1 SOFTCTLONLY	RO	0x1
6	portc_single_ctl	The value of this register is derived from the GPIO_PORTC_SINGLE_CTL configuration parameter. 0 = False 1 = True **Value** **Description** 0x1 SOFTCTLONLY	RO	0x1
5	portb_single_ctl	The value of this register is derived from the GPIO_PORTB_SINGLE_CTL configuration parameter. 0 = False 1 = True **Value** **Description** 0x1 SOFTCTLONLY	RO	0x1

Bit	Name	Description	Access	Reset
4	porta_single_ctl	The value of this register is derived from the GPIO_PORTA_SINGLE_CTL configuration parameter. 0 = False 1 = True **Value** **Description** 0x1 SOFTCTLONLY	RO	0x1
3:2	num_ports	The value of this register is derived from the GPIO_NUM_PORT configuration parameter. 0x0 =1 0x1 = 2 0x2 = 3 0x3 = 4 **Value** **Description** 0x0 ONEPORTA	RO	0x0
1:0	apb_data_width	The value of this register is derived from the GPIO_APB_DATA_WIDTH configuration parameter. 0x0 = 8 bits 0x1 = 16 bits 0x2 = 32 bits 0x3 = Reserved **Value** **Description** 0x2 WIDTH32BITS	RO	0x2

Document Revision History

Table 22-3: Document Revision History

Date	Version	Changes
October 2016	2016.10.28	Maintenance release.
May 2016	2016.05.03	Maintenance release.
November 2015	2015.11.02	Maintenance release.
May 2015	2015.05.04	Maintenance release.

Date	Version	Changes
December 2014	2014.12.15	• Maintenance release. • Added *Taking the GPIO Out of Reset* section.
August 2014	2014.08.18	Initial release.

Timer 23

2016.10.28

a10_5v4 Subscribe Send Feedback

The hard processor system (HPS) provides four 32-bit general-purpose timers connected to the level 4 (L4) peripheral bus.The timers optionally generate an interrupt when the 32-bit binary count-down timer reaches zero. The timers are instances of the Synopsys DesignWare APB Timers (DW_apb_timers) peripheral. (56)

Related Information

Cortex-A9 Microprocessor Unit Subsystem on page 9-1
The MPU subsystem provides additional timers. For more information about the timers in the MPU, refer to the *Cortex-A9 MPU* chapter.

Features of the Timer

- Supports interrupt generation
- Supports free-running mode
- Supports user-defined count mode

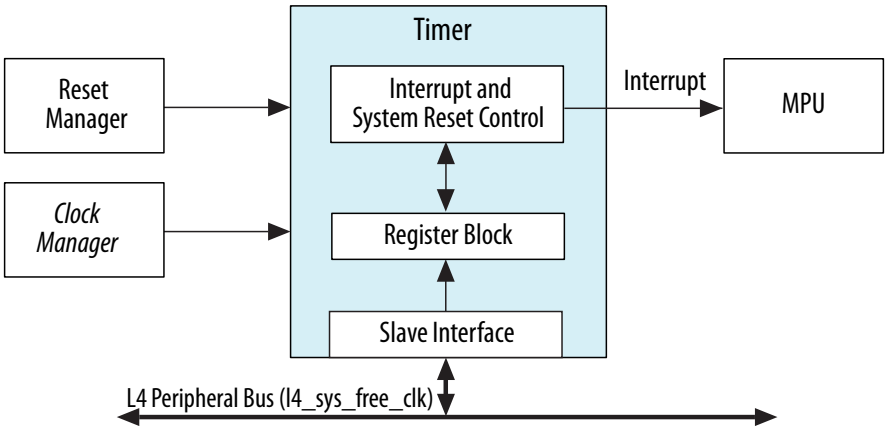
Timer Block Diagram and System Integration

Each timer includes a slave interface for control and status register (CSR) access, a register block, and a programmable 32-bit down counter that generates interrupts on reaching zero. The timer operates on a single clock domain driven by the clock manager.

(56) Portions © 2016 Synopsys, Inc. Used with permission. All rights reserved. Synopsys & DesignWare are registered trademarks of Synopsys, Inc. All documentation is provided "as is" and without any warranty. Synopsys expressly disclaims any and all warranties, express, implied, or otherwise, including the implied warranties of merchantability, fitness for a particular purpose, and non-infringement, and any warranties arising out of a course of dealing or usage of trade.

Paragraphs marked with the dagger (†) symbol are Synopsys Proprietary. Used with permission.

Figure 23-1: Timer Block Diagram



Functional Description of the Timer

The 32-bit timer counts down from a programmed value and generates an interrupt when the count reaches zero. The timer has an independent clock input connected to the system clock signal or to an external clock source. †

The timer supports the following modes of operation:

- Free-running mode—decrementing from the maximum value (0xFFFFFFFF). Reloads maximum value upon reaching zero.
- User-defined count mode—generates a periodic interrupt. Decrements from the user-defined count value loaded from the timer1 load count register (`timer1loadcount`). Reloads the user-defined count upon reaching zero.

The initial value for the timer (that is, the value from which it counts down) is loaded into the timer by the `timer1loadcount` register. The following events can cause a timer to load the initial count from the `timer1loadcount` register: †

- Timer is enabled after being reset or disabled
- Timer counts down to 0

Clocks

Table 23-1: Timer Clock Characteristics

Timer		System Clock	Notes
System timer 0	sys_timer0	l4_sys_free_clk	_
System timer 1	sys_timer1		
SP timer 0	sp_timer0	l4_sp_clk	Timer must be disabled if clock frequency changes
SP timer 1	sp_timer1		

The timers above are labeled according to the clock they receive. The system timers are connected to the L4_SYS bus and clocked by the l4_sys_free_clk. The SP timers are connected to the L4_SP bus and clocked by l4_sp_clk.

SP timer 0 and SP timer 1 must be disabled before l4_sp_clk is changed to another frequency. You can then re-enable the timer once the clock frequency change takes effect.

Related Information

Clock Manager on page 2-1
For more information about clock performance, refer to the *Clock Manager* chapter.

Resets

The timers are reset by a cold or warm reset. Resetting the timers produces the following results in the following order:

1. The timer is disabled.
2. The interrupt is enabled.
3. The timer enters free-running mode.
4. The timer count load register value is set to zero.

Interrupts

The timer1 interrupt status (timer1intstat) and timer1 end of interrupt (timer1eoi) registers handle the interrupts. The timer1intstat register allows you to read the status of the interrupt. Reading from the timer1eoi register clears the interrupt. †

The timer1 control register (timer1controlreg) contains the timer1 interrupt mask bit (timer1_interrupt_mask) to mask the interrupt. In both the free-running and user-defined count modes of operation, the timer generates an interrupt signal when the timer count reaches zero and the interrupt mask bit of the control register is high.

If the timer interrupt is set, then it is cleared when the timer is disabled.

Timer Programming Model

Initialization

To initialize the timer, perform the following steps: †

1. Initialize the timer through the `timer1controlreg` register: †
   - Disable the timer by writing a 0 to the timer1 enable bit (`timer1_enable`) of the `timer1controlreg` register. †

   Note: Before writing to a timer1 load count register (`timer1loadcount`), you must disable the timer by writing a 0 to the `timer1_enable` bit of the `timer1controlreg` register to avoid potential synchronization problems. †
   - Program the timer mode—user-defined count or free-running—by writing a 0 or 1, respectively, to the timer1 mode bit (`timer1_mode`) of the `timer1controlreg` register. †
   - Set the interrupt mask as either masked or not masked by writing a 1 or 0, respectively, to the `timer1_interrupt_mask` bit of the `timer1controlreg` register. †
2. Load the timer counter value into the `timer1loadcount` register. †
3. Enable the timer by writing a 1 to the `timer1_enable` bit of the `timer1controlreg` register. †

Enabling the Timer

When a timer transitions to the enabled state, the current value of `timer1loadcount` register is loaded into the timer counter. †

1. To enable the timer, write a 1 to the `timer1_enable` bit of the `timer1controlreg` register.

Disabling the Timer

When the timer enable bit is cleared to 0, the timer counter and any associated registers in the timer clock domain, are asynchronously reset. †

1. To disable the timer, write a 0 to the `timer1_enable` bit. †

Loading the Timer Countdown Value

When a timer counter is enabled after being reset or disabled, the count value is loaded from the `timer1loadcount` register; this occurs in both free-running and user-defined count modes. †

When a timer counts down to 0, it loads one of two values, depending on the timer operating mode: †

- User-defined count mode—timer loads the current value of the `timer1loadcount` register. Use this mode if you want a fixed, timed interrupt. Designate this mode by writing a 1 to the `timer1_mode` bit of the `timer1controlreg` register. †
- Free-running mode—timer loads the maximum value (0xFFFFFFFF). The timer max count value allows for a maximum amount of time to reprogram or disable the timer before another interrupt occurs. Use this mode if you want a single timed interrupt. Enable this mode by writing a 0 to the `timer1_mode` bit of the `timer1controlreg` register. †

Servicing Interrupts

Clearing the Interrupt

An active timer interrupt can be cleared in two ways.

1. If you clear the interrupt at the same time as the timer reaches 0, the interrupt remains asserted. This action happens because setting the timer interrupt takes precedence over clearing the interrupt. †
2. To clear an active timer interrupt, read the `timer1eoi` register or disable the timer. When the timer is enabled, its interrupt remains asserted until it is cleared by reading the `timer1eoi` register. †

Checking the Interrupt Status

You can query the interrupt status of the timer without clearing its interrupt.

1. To check the interrupt status, read the `timer1intstat` register. †

Masking the Interrupt

The timer interrupt can be masked using the `timer1controlreg` register.

To mask an interrupt, write a 1 to the `timer1_interrupt_mask` bit of the `timer1controlreg` register. †

Timer Address Map and Register Definitions

For complete HPS address map and register definitions, refer to the **Arria 10 HPS Address Map and Register Definitions**.

timer_sys Address Map

Module Instance	Base Address	End Address
`i_timer_sys_0_timer`	`0xFFD00000`	`0xFFD000FF`
`i_timer_sys_1_timer`	`0xFFD00100`	`0xFFD001FF`

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
`timer1loadcount` on page 23-11	`0x0`	32	RW	`0x0`	Name: Timer1 Load Count Register Size: 8-32 bits Address Offset: 0x00 Read/Write Access: Read/Write

Register	Offset	Width	Access	Reset Value	Description
timer1currentval on page 23-11	0x4	32	RO	0x0	Name: Timer1 Current Value Size: 8-32 bits Address Offset: 4 Read/Write Access: Read
timer1controlreg on page 23-12	0x8	32	RW	0x0	Name: Timer1 Control Register Size: 3 bits Address Offset: 8 Read/Write Access: Read/Write This register controls enabling, operating mode (free-running or defined-count), and interrupt mask of Timer1. You can program each Timer1ControlReg to enable or disable a specific timer and to control its mode of operation.
timer1eoi on page 23-14	0xC	32	RO	0x0	Name: Timer1 End-of-Interrupt Register Size: 1 bit Address Offset: 12 Read/Write Access: Read
timer1intstat on page 23-15	0x10	32	RO	0x0	Name: Timer1 Interrupt Status Register Size: 1 bit Address Offset: 16 Read/Write Access: Read
timersintstat on page 23-16	0xA0	32	RO	0x0	Name: Timers Interrupt Status Register Size: 1-9 bits Address Offset: 0xa0 Read/Write Access: Read

Register	Offset	Width	Access	Reset Value	Description
timerseoi on page 23-17	0xA4	32	RO	0x0	Name: Timers End-of-Interrupt Register Size: 1-9 bits Address Offset: 0xa4 Read/Write Access: Read
timersrawintstat on page 23-18	0xA8	32	RO	0x0	Name: Timers Raw Interrupt Status Register Size: 1-9 bits Address Offset: 0xa8 Read/Write Access: Read
timerscompversion on page 23-19	0xAC	32	RO	0x3230382A	Name: Timers Component Version Size: 32 bits Address Offset: 0xac Read/Write Access: Read

timer_sys Summary

Module Instance	Base Address
i_timer_sys_0_timer	0xFFD00000
i_timer_sys_1_timer	0xFFD00100

Register Address Offset	Bit Fields															
i_timer_sys_0_timer																
timer1loadcount 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	timer1loadcount RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	timer1loadcount RW 0x0															

Register Address Offset	Bit Fields															
timer1currentval 0x4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	timer1currentval RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	timer1currentval RO 0x0															
timer1controlreg 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													timer1_interrupt_mask RW 0x0	timer1_mode RW 0x0	timer1_enable RW 0x0
timer1eoi 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															timer1eoi RO 0x0
timer1intstat 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															timer1intstat RO 0x0
timersint-stat 0xA0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															timersintstat RO 0x0
timerseoi 0xA4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															timerseoi RO 0x0

Register Address Offset	Bit Fields															
timersra-wintstat 0xA8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															timersrawintstat RO 0x0
timerscomp-version 0xAC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	timerscompversion RO 0x3230382A															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	timerscompversion RO 0x3230382A															
i_timer_sys_1_timer																
timer1loadcount 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	timer1loadcount RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	timer1loadcount RW 0x0															
timer1currentval 0x4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	timer1currentval RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	timer1currentval RO 0x0															
timer1controlreg 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													timer1_interrupt_mask RW 0x0	timer1_mode RW 0x0	timer1_enable RW 0x0

Register Address Offset	Bit Fields															
timer1eoi 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															timer1eoi RO 0x0
timer1intstat 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															timer1intstat RO 0x0
timersint-stat 0xA0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															timersintstat RO 0x0
timerseoi 0xA4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															timerseoi RO 0x0
timersra-wintstat 0xA8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															timersrawintstat RO 0x0
timerscomp-version 0xAC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	timerscompversion RO 0x3230382A															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	timerscompversion RO 0x3230382A															

timer1loadcount

```
        Name: Timer1 Load Count Register
Size: 8-32 bits
Address Offset: 0x00
Read/Write Access: Read/Write
```

Module Instance	Base Address	Register Address
i_timer_sys_0_timer	0xFFD00000	0xFFD00000
i_timer_sys_1_timer	0xFFD00100	0xFFD00100

Offset: 0x0

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
timer1loadcount RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
timer1loadcount RW 0x0															

timer1loadcount Fields

Bit	Name	Description	Access	Reset
31:0	timer1loadcount	Value to be loaded into Timer1. This is the value from which counting commences. Any value written to this register is loaded into the associated timer.	RW	0x0

timer1currentval

```
        Name: Timer1 Current Value
Size: 8-32 bits
Address Offset: 4
Read/Write Access: Read
```

Module Instance	Base Address	Register Address
i_timer_sys_0_timer	0xFFD00000	0xFFD00004
i_timer_sys_1_timer	0xFFD00100	0xFFD00104

Offset: 0x4

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
timer1currentval RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
timer1currentval RO 0x0															

timer1currentval Fields

Bit	Name	Description	Access	Reset
31:0	timer1currentval	Current Value of Timer1. This register is supported only when timer_1_clk is synchronous to pclk. Reading this register when using independent clocks results in an undefined value.	RO	0x0

timer1controlreg

```
        Name: Timer1 Control Register
  Size: 3 bits
  Address Offset: 8
  Read/Write Access: Read/Write
  This register controls enabling, operating mode (free-running or
defined-count), and interrupt mask of
  Timer1. You can program each Timer1ControlReg to enable or disable
a specific timer and to control
  its mode of operation.
```

Module Instance	Base Address	Register Address
i_timer_sys_0_timer	0xFFD00000	0xFFD00008
i_timer_sys_1_timer	0xFFD00100	0xFFD00108

Offset: 0x8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													timer_1_interrupt_mask RW 0x0	timer_1_mode RW 0x0	timer1_enable RW 0x0

timer1controlreg Fields

Bit	Name	Description	Access	Reset
2	timer1_interrupt_mask	Timer interrupt mask for Timer1. 0: not masked 1: masked Value: Description 0x0: NOTMASKED 0x1: MASKED	RW	0x0
1	timer1_mode	Timer mode for Timer1. 0: free-running mode 1: user-defined count mode NOTE: You must set the Timer1LoadCount register to all 1s before enabling the timer in free-running mode. Value: Description 0x0: FREERUN 0x1: USEDEF	RW	0x0

Bit	Name	Description	Access	Reset
0	timer1_enable	Timer enable bit for Timer1. 0: disable 1: enable Value: Description 0x0: DISABLED 0x1: ENABLED	RW	0x0

timer1eoi

Name: Timer1 End-of-Interrupt Register
Size: 1 bit
Address Offset: 12
Read/Write Access: Read

Module Instance	Base Address	Register Address
i_timer_sys_0_timer	0xFFD00000	0xFFD0000C
i_timer_sys_1_timer	0xFFD00100	0xFFD0010C

Offset: 0xC

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															timer1eoi RO 0x0

timer1eoi Fields

Bit	Name	Description	Access	Reset
0	timer1eoi	Reading from this register returns all zeroes (0) and clears the interrupt from Timer1.	RO	0x0

timer1intstat

```
Name: Timer1 Interrupt Status Register
Size: 1 bit
Address Offset: 16
Read/Write Access: Read
```

Module Instance	Base Address	Register Address
i_timer_sys_0_timer	0xFFD00000	0xFFD00010
i_timer_sys_1_timer	0xFFD00100	0xFFD00110

Offset: 0x10

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															timer1intstat RO 0x0

timer1intstat Fields

Bit	Name	Description	Access	Reset
0	timer1intstat	Contains the interrupt status for Timer1. Value: Description 0x0: INACTIVE 0x1: ACTIVE	RO	0x0

timersintstat

```
Name: Timers Interrupt Status Register
Size: 1-9 bits
Address Offset: 0xa0
Read/Write Access: Read
```

Module Instance	Base Address	Register Address
i_timer_sys_0_timer	0xFFD00000	0xFFD000A0
i_timer_sys_1_timer	0xFFD00100	0xFFD001A0

Offset: 0xA0

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															timersintstat RO 0x0

timersintstat Fields

Bit	Name	Description	Access	Reset
0	timersintstat	Contains the interrupt status of all timers in the component. If a bit of this register is 0, then the corresponding timer interrupt is not active and the corresponding interrupt could be on either the timer_intr bus or the timer_intr_n bus, depending on the interrupt polarity you have chosen. Similarly, if a bit of this register is 1, then the corresponding interrupt bit has been set in the relevant interrupt bus. In both cases, the status reported is the status after the interrupt mask has been applied. Reading from this register does not clear any active interrupts: 0 = either timer_intr or timer_intr_n is not active after masking 1 = either timer_intr or timer_intr_n is active after masking. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0

timerseoi

```
        Name: Timers End-of-Interrupt Register
Size: 1-9 bits
Address Offset: 0xa4
Read/Write Access: Read
```

Module Instance	Base Address	Register Address
i_timer_sys_0_timer	0xFFD00000	0xFFD000A4
i_timer_sys_1_timer	0xFFD00100	0xFFD001A4

Offset: 0xA4

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															timerseoi RO 0x0

timerseoi Fields

Bit	Name	Description	Access	Reset
0	timerseoi	Reading this register returns all zeroes (0) and clears all active interrupts.	RO	0x0

timersrawintstat

```
          Name: Timers Raw Interrupt Status Register
Size: 1-9 bits
Address Offset: 0xa8
Read/Write Access: Read
```

Module Instance	Base Address	Register Address
i_timer_sys_0_timer	0xFFD00000	0xFFD000A8
i_timer_sys_1_timer	0xFFD00100	0xFFD001A8

Offset: 0xA8

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															timersrawintstat RO 0x0

timersrawintstat Fields

Bit	Name	Description	Access	Reset
0	timersrawintstat	The register contains the unmasked interrupt status of all timers in the component. 0 = either timer_intr or timer_intr_n is not active prior to masking 1 = either timer_intr or timer_intr_n is active prior to masking. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0

timerscompversion

```
        Name: Timers Component Version
Size: 32 bits
Address Offset: 0xac
Read/Write Access: Read
```

Module Instance	Base Address	Register Address
i_timer_sys_0_timer	0xFFD00000	0xFFD000AC
i_timer_sys_1_timer	0xFFD00100	0xFFD001AC

Offset: 0xAC

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
timerscompversion RO 0x3230382A															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
timerscompversion RO 0x3230382A															

timerscompversion Fields

Bit	Name	Description	Access	Reset
31:0	timerscompversion	Current revision number of the DW_apb_timers component.	RO	0x3230382A

timer Address Map

Module Instance	Base Address	End Address
i_timer_sp_0_timer	0xFFC02700	0xFFC027FF
i_timer_sp_1_timer	0xFFC02800	0xFFC028FF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
timer1loadcount on page 23-25	0x0	32	RW	0x0	Name: Timer1 Load Count Register Size: 8-32 bits Address Offset: 0x00 Read/Write Access: Read/Write
timer1currentval on page 23-26	0x4	32	RO	0x0	Name: Timer1 Current Value Size: 8-32 bits Address Offset: 4 Read/Write Access: Read

Register	Offset	Width	Access	Reset Value	Description
timer1controlreg on page 23-27	0x8	32	RW	0x0	Name: Timer1 Control Register Size: 3 bits Address Offset: 8 Read/Write Access: Read/Write This register controls enabling, operating mode (free-running or defined-count), and interrupt mask of Timer1. You can program each Timer1ControlReg to enable or disable a specific timer and to control its mode of operation.
timer1eoi on page 23-29	0xC	32	RO	0x0	Name: Timer1 End-of-Interrupt Register Size: 1 bit Address Offset: 12 Read/Write Access: Read
timer1intstat on page 23-29	0x10	32	RO	0x0	Name: Timer1 Interrupt Status Register Size: 1 bit Address Offset: 16 Read/Write Access: Read
timersintstat on page 23-30	0xA0	32	RO	0x0	Name: Timers Interrupt Status Register Size: 1-9 bits Address Offset: 0xa0 Read/Write Access: Read
timerseoi on page 23-32	0xA4	32	RO	0x0	Name: Timers End-of-Interrupt Register Size: 1-9 bits Address Offset: 0xa4 Read/Write Access: Read

Register	Offset	Width	Access	Reset Value	Description
timersrawintstat on page 23-33	0xA8	32	RO	0x0	Name: Timers Raw Interrupt Status Register Size: 1-9 bits Address Offset: 0xa8 Read/Write Access: Read
timerscompversion on page 23-34	0xAC	32	RO	0x3230382A	Name: Timers Component Version Size: 32 bits Address Offset: 0xac Read/Write Access: Read

timer Summary

Module Instance	Base Address
i_timer_sp_0_timer	0xFFC02700
i_timer_sp_1_timer	0xFFC02800

Register Address Offset	Bit Fields															
i_timer_sp_0_timer																
timer1loadcount 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	timer1loadcount RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	timer1loadcount RW 0x0															
timer1currentval 0x4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	timer1currentval RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	timer1currentval RO 0x0															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
timer1controlreg 0x8	Reserved													timer1_interrupt_mask RW 0x0	timer1_mode RW 0x0	timer1_enable RW 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
timer1eoi 0xC	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															timer1eoi RO 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
timer1intstat 0x10	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															timer1intstat RO 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
timersint-stat 0xA0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															timersintstat RO 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
timerseoi 0xA4	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															timerseoi RO 0x0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
timersra-wintstat 0xA8	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															timersrawintstat RO 0x0

Register Address Offset	Bit Fields															
timerscomp-version 0xAC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	timerscompversion RO 0x3230382A															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	timerscompversion RO 0x3230382A															
i_timer_sp_1_timer																
timer1loadcount 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	timer1loadcount RW 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	timer1loadcount RW 0x0															
timer1currentval 0x4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	timer1currentval RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	timer1currentval RO 0x0															
timer1controlreg 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													timer1_interrupt_mask RW 0x0	timer1_mode RW 0x0	timer1_enable RW 0x0
timer1eoi 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															timer1eoi RO 0x0

Register Address Offset	Bit Fields															
timer1intstat 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															timer1intstat RO 0x0
timersint-stat 0xA0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															timersintstat RO 0x0
timerseoi 0xA4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															timerseoi RO 0x0
timersra-wintstat 0xA8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															timersrawintstat RO 0x0
timerscomp-version 0xAC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	timerscompversion RO 0x3230382A															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	timerscompversion RO 0x3230382A															

timer1loadcount

```
        Name: Timer1 Load Count Register
Size: 8-32 bits
Address Offset: 0x00
Read/Write Access: Read/Write
```

Module Instance	Base Address	Register Address
i_timer_sp_0_timer	0xFFC02700	0xFFC02700
i_timer_sp_1_timer	0xFFC02800	0xFFC02800

Offset: 0x0

Access: RW

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
timer1loadcount RW 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
timer1loadcount RW 0x0															

timer1loadcount Fields

Bit	Name	Description	Access	Reset
31:0	timer1loadcount	Value to be loaded into Timer1. This is the value from which counting commences. Any value written to this register is loaded into the associated timer.	RW	0x0

timer1currentval

```
        Name: Timer1 Current Value
Size: 8-32 bits
Address Offset: 4
Read/Write Access: Read
```

Module Instance	Base Address	Register Address
i_timer_sp_0_timer	0xFFC02700	0xFFC02704
i_timer_sp_1_timer	0xFFC02800	0xFFC02804

Offset: 0x4

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
timer1currentval RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
timer1currentval RO 0x0															

timer1currentval Fields

Bit	Name	Description	Access	Reset
31:0	timer1currentval	Current Value of Timer1. This register is supported only when timer_1_clk is synchronous to pclk. Reading this register when using independent clocks results in an undefined value.	RO	0x0

timer1controlreg

```
        Name: Timer1 Control Register
  Size: 3 bits
  Address Offset: 8
  Read/Write Access: Read/Write
  This register controls enabling, operating mode (free-running or
defined-count), and interrupt mask of
  Timer1. You can program each Timer1ControlReg to enable or disable
a specific timer and to control
  its mode of operation.
```

Module Instance	Base Address	Register Address
i_timer_sp_0_timer	0xFFC02700	0xFFC02708
i_timer_sp_1_timer	0xFFC02800	0xFFC02808

Offset: 0x8

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													timer 1_ interrupt_ mask RW 0x0	timer 1_ mode RW 0x0	timer1_ enable RW 0x0

timer1controlreg Fields

Bit	Name	Description	Access	Reset
2	timer1_interrupt_mask	Timer interrupt mask for Timer1. 0: not masked 1: masked **Value** **Description** 0x0 NOTMASKED 0x1 MASKED	RW	0x0
1	timer1_mode	Timer mode for Timer1. 0: free-running mode 1: user-defined count mode NOTE: You must set the Timer1LoadCount register to all 1s before enabling the timer in free-running mode. **Value** **Description** 0x0 FREERUN 0x1 USEDEF	RW	0x0
0	timer1_enable	Timer enable bit for Timer1. 0: disable 1: enable **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0

timer1eoi

```
Name: Timer1 End-of-Interrupt Register
Size: 1 bit
Address Offset: 12
Read/Write Access: Read
```

Module Instance	Base Address	Register Address
i_timer_sp_0_timer	0xFFC02700	0xFFC0270C
i_timer_sp_1_timer	0xFFC02800	0xFFC0280C

Offset: 0xC

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															timer1eoi RO 0x0

timer1eoi Fields

Bit	Name	Description	Access	Reset
0	timer1eoi	Reading from this register returns all zeroes (0) and clears the interrupt from Timer1.	RO	0x0

timer1intstat

```
Name: Timer1 Interrupt Status Register
Size: 1 bit
Address Offset: 16
Read/Write Access: Read
```

Module Instance	Base Address	Register Address
i_timer_sp_0_timer	0xFFC02700	0xFFC02710
i_timer_sp_1_timer	0xFFC02800	0xFFC02810

Offset: 0x10

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															timer1intstat RO 0x0

timer1intstat Fields

Bit	Name	Description	Access	Reset
0	timer1intstat	Contains the interrupt status for Timer1. Value: Description 0x0: INACTIVE 0x1: ACTIVE	RO	0x0

timersintstat

```
        Name: Timers Interrupt Status Register
Size: 1-9 bits
Address Offset: 0xa0
Read/Write Access: Read
```

Module Instance	Base Address	Register Address
i_timer_sp_0_timer	0xFFC02700	0xFFC027A0
i_timer_sp_1_timer	0xFFC02800	0xFFC028A0

Offset: 0xA0

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															timersintstat RO 0x0

timersintstat Fields

Bit	Name	Description	Access	Reset
0	timersintstat	Contains the interrupt status of all timers in the component. If a bit of this register is 0, then the corresponding timer interrupt is not active and the corresponding interrupt could be on either the timer_intr bus or the timer_intr_n bus, depending on the interrupt polarity you have chosen. Similarly, if a bit of this register is 1, then the corresponding interrupt bit has been set in the relevant interrupt bus. In both cases, the status reported is the status after the interrupt mask has been applied. Reading from this register does not clear any active interrupts: 0 = either timer_intr or timer_intr_n is not active after masking 1 = either timer_intr or timer_intr_n is active after masking. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0

timerseoi

Name: Timers End-of-Interrupt Register
Size: 1-9 bits
Address Offset: 0xa4
Read/Write Access: Read

Module Instance	Base Address	Register Address
i_timer_sp_0_timer	0xFFC02700	0xFFC027A4
i_timer_sp_1_timer	0xFFC02800	0xFFC028A4

Offset: 0xA4

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															timerseoi RO 0x0

timerseoi Fields

Bit	Name	Description	Access	Reset
0	timerseoi	Reading this register returns all zeroes (0) and clears all active interrupts.	RO	0x0

timersrawintstat

```
Name: Timers Raw Interrupt Status Register
Size: 1-9 bits
Address Offset: 0xa8
Read/Write Access: Read
```

Module Instance	Base Address	Register Address
i_timer_sp_0_timer	0xFFC02700	0xFFC027A8
i_timer_sp_1_timer	0xFFC02800	0xFFC028A8

Offset: 0xA8

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															timersrawintstat RO 0x0

timersrawintstat Fields

Bit	Name	Description	Access	Reset
0	timersrawintstat	The register contains the unmasked interrupt status of all timers in the component. 0 = either timer_intr or timer_intr_n is not active prior to masking 1 = either timer_intr or timer_intr_n is active prior to masking. **Value** **Description** 0x0 INACTIVE 0x1 ACTIVE	RO	0x0

timerscompversion

```
        Name: Timers Component Version
Size: 32 bits
Address Offset: 0xac
Read/Write Access: Read
```

Module Instance	Base Address	Register Address
i_timer_sp_0_timer	0xFFC02700	0xFFC027AC
i_timer_sp_1_timer	0xFFC02800	0xFFC028AC

Offset: 0xAC

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
timerscompversion RO 0x3230382A															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
timerscompversion RO 0x3230382A															

timerscompversion Fields

Bit	Name	Description	Access	Reset
31:0	timerscompversion	Current revision number of the DW_apb_timers component.	RO	0x3230382A

Document Revision History

Table 23-2: Document Revision History

Date	Version	Changes
October 2016	2016.10.28	Maintenance release.
May 2016	2016.05.27	Maintenance release.
May 2016	2016.05.03	Maintenance release.
November 2015	2015.11.02	Maintenance release.
May 2015	2015.05.04	Maintenance release.
December 2014	2014.12.15	Maintenance release.
August 2014	2014.08.18	Initial release.

Watchdog Timer 24

2016.10.28

a10_5v4 Subscribe Send Feedback

The watchdog timers are peripherals you can use to recover from system lockup that might be caused by software or system related issues. The hard processor system (HPS) provides two programmable watchdog timers, which are connected to the level 4 (L4) peripheral bus. The watchdog timers are instances of the Synopsys DesignWare APB Watchdog Timer (DW_apb_wdt) peripheral. (57)

The microprocessor unit (MPU) subsystem provides two additional watchdog timers.

Related Information

Cortex-A9 Microprocessor Unit Subsystem on page 9-1
For more information about the watchdog timers in the MPU, refer to *Cortex A9 Microprocessor Unit Subsystem* chapter.

Features of the Watchdog Timer

The following list describes the features of the watchdog timer:

- Programmable 32-bit timeout range
- Timer counts down from a preset value to zero, then performs one of the following user-configurable operations:
  - Generates a system reset †
  - Generates an interrupt, restarts the timer, and if the timer is not cleared before a second timeout occurs, generates a system reset
- Dual programmable timeout period, used when the time to wait after the first start is different than that required for subsequent restarts †
- Prevention of accidental restart of the watchdog counter †
- Prevention of accidental disabling of the watchdog counter †
- Pause mode for debugging

(57) Portions © 2016 Synopsys, Inc. Used with permission. All rights reserved. Synopsys & DesignWare are registered trademarks of Synopsys, Inc. All documentation is provided "as is" and without any warranty. Synopsys expressly disclaims any and all warranties, express, implied, or otherwise, including the implied warranties of merchantability, fitness for a particular purpose, and non-infringement, and any warranties arising out of a course of dealing or usage of trade.

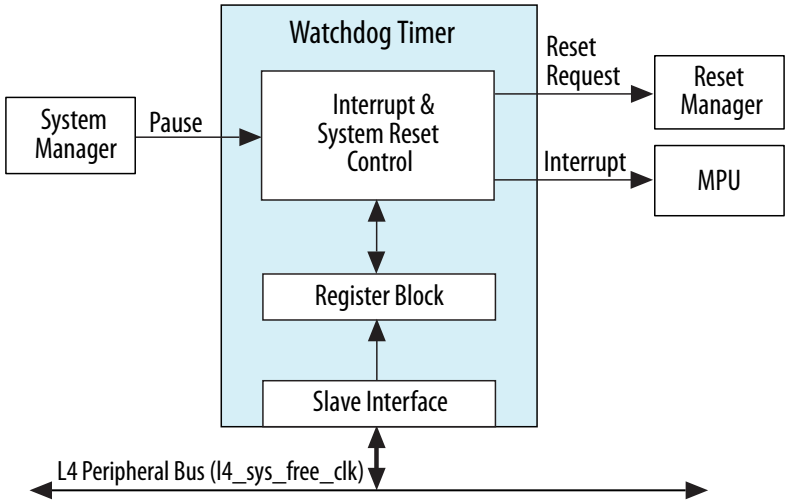
†Paragraphs marked with the dagger (†) symbol are Synopsys Proprietary. Used with permission.

Watchdog Timer Block Diagram and System Integration

Each watchdog timer consists of a slave interface for control and status register (CSR) access, a register block, and a 32-bit down counter that operates on the slave interface clock (`l4_sys_free_clk`). A pause input, driven by the system manager, optionally pauses the counter when a CPU is being debugged.

The watchdog timer drives an interrupt request to the MPU and a reset request to the reset manager.

Figure 24-1: Watchdog Timer Block Diagram



Related Information

- **Reset Manager** on page 3-1
  For more information, refer to the *Reset Manager* chapter.
- **Cortex-A9 Microprocessor Unit Subsystem** on page 9-1
  For more information about the watchdog timers in the MPU, refer to *Cortex A9 Microprocessor Unit Subsystem* chapter.

Functional Description of the Watchdog Timer

Watchdog Timer Counter

Each watchdog timer is a programmable, little-endian down counter that decrements by one on each clock cycle. The watchdog timer supports 16 fixed timeout period values, Software chooses which timeout periods are desired. A timeout period is 2<*n*> `l4_sys_free_clk` clock periods, where *n* is an integer from 16 to 31 inclusive.

Software must regularly restart the timer (which reloads the counter with the restart timeout period value) to indicate that the system is functioning normally. Software can reload the counter at any time by writing to the restart register. If the counter reaches zero, the watchdog timer has timed out, indicating an unrecoverable error has occurred and a system reset is needed.

Software configures the watchdog timer to one of the following output response modes:

- On timeout, generate a reset request.
- On timeout, assert an interrupt request and restart the watchdog timer. Software must service the interrupt and reset the watchdog timer before a second timeout occurs. Otherwise, generate a reset request.

If a restart occurs at the same time the watchdog counter reaches zero, an interrupt is not generated.

Note: After the watchdog timer reaches zero and generates a reset or interrupt, the counter resets and continues to count.

Related Information

- **Watchdog Timer Clocks** on page 24-3
- **Setting the Timeout Period Values** on page 24-4
- **Selecting the Output Response Mode** on page 24-5
- **Reloading a Watchdog Counter** on page 24-5

Watchdog Timer Pause Mode

The watchdog timers can be paused during debugging. The watchdog timer pause mode is controlled by the system manager. The following options are available:

- Pause the timer while either CPU0 or CPU1 is in debug mode
- Pause the timer while only CPU1 is in debug mode
- Pause the timer while only CPU0 is in debug mode
- Do not pause the timer

When pause mode is enabled, the system manager pauses the watchdog timer while debugging. When pause mode is disabled, the watchdog timer runs while debugging.

At reset, the watchdog pausing feature is enabled for both CPUs by default.

Related Information

Pausing a Watchdog Timer on page 24-5

Watchdog Timer Clocks

Each watchdog timer is connected to the `l4_sys_free_clk` clock so that timer operation is not dependent on the phase-locked loops (PLLs) in the clock manager and so that it is always running. This independence allows recovery from software that inadvertently programs the PLLs in the clock manager incorrectly.

Table 24-1: Watchdog Timer Clocks

Timer	System Clock
`watchdog0`	`l4_sys_free_clk`
`watchdog1`	`l4_sys_free_clk`

Table 24-2: Watchdog Timer Clocks

Timer	System Clock
watchdog0	l4_sys_free_clk
watchdog1	l4_sys_free_clk
watchdog2	l4_sys_free_clk
watchdog3	l4_sys_free_clk

Related Information

Clock Manager on page 2-1
For more information, refer to the *Clock Manager* chapter.

Watchdog Timer Resets

Watchdog timers are reset by a cold or warm reset from the reset manager, and are disabled when exiting reset. †

Related Information

Reset Manager on page 3-1
For more information, refer to the *Reset Manager* chapter.

Taking the Watchdog Timer Out of Reset

When a cold or warm reset is issued in the HPS, the reset manager resets this module and holds it in reset until software releases it.

After the Cortex-A9 MPCore CPU boots, it can deassert the reset signal by clearing the appropriate bits in the reset manager's corresponding reset register. For details about reset registers, refer to "Module Reset Signals".

Watchdog Timer Programming Model

Setting the Timeout Period Values

The watchdog timers have a dual timeout period. The counter uses the initial start timeout period value the first the timer is started. All subsequent restarts use the restart timeout period. The valid values are $2^{(16+<i>)}$ – 1 clock cycles, where i is an integer from 0 to 15. To set the programmable timeout periods, perform the following actions in no specific order:

Note: Set the timeout values before enabling the timer.

- To set the initial start timeout period, write i to the timeout period for the initialization field (`top_init`) of the watchdog timeout range register (`wdt_torr`).
- To set the restart timeout period, write i to the timeout period field (`top`) of the `wdt_torr` register

Selecting the Output Response Mode

The watchdog timers have two output response modes. To select the desired mode, perform one of the following actions:

- To generate a system reset request when a timeout occurs, write 0 to the output response mode bit (`rmod`) of the watchdog timer control register (`wdt_cr`).
- To generate an interrupt and restart the timer when a timeout occurs, write 1 to the `rmod` field of the `wdt_cr` register.

If a restart occurs at the same time the watchdog counter reaches zero, a system reset is not generated. †

Related Information

Watchdog Timer Counter on page 24-2

Enabling and Initially Starting a Watchdog Timer

To enable and start a watchdog timer, write the value 1 to the watchdog timer enable bit (`wdt_en`) of the `wdt_cr` register.

Reloading a Watchdog Counter

To reload a watchdog counter, write the value 0x76 to the counter restart register (`wdt_crr`). This unique 8-bit value is used as a safety feature to prevent accidental restarts.

Pausing a Watchdog Timer

Pausing the watchdog timers is controlled by the L4 watchdog debug register (`wddbg`) in the system manager.

Related Information

Features of the System Manager on page 5-1
For more information, refer to the *System Manager* chapter.

Disabling and Stopping a Watchdog Timer

The watchdog timers are disabled and stopped by resetting them from the reset manager.

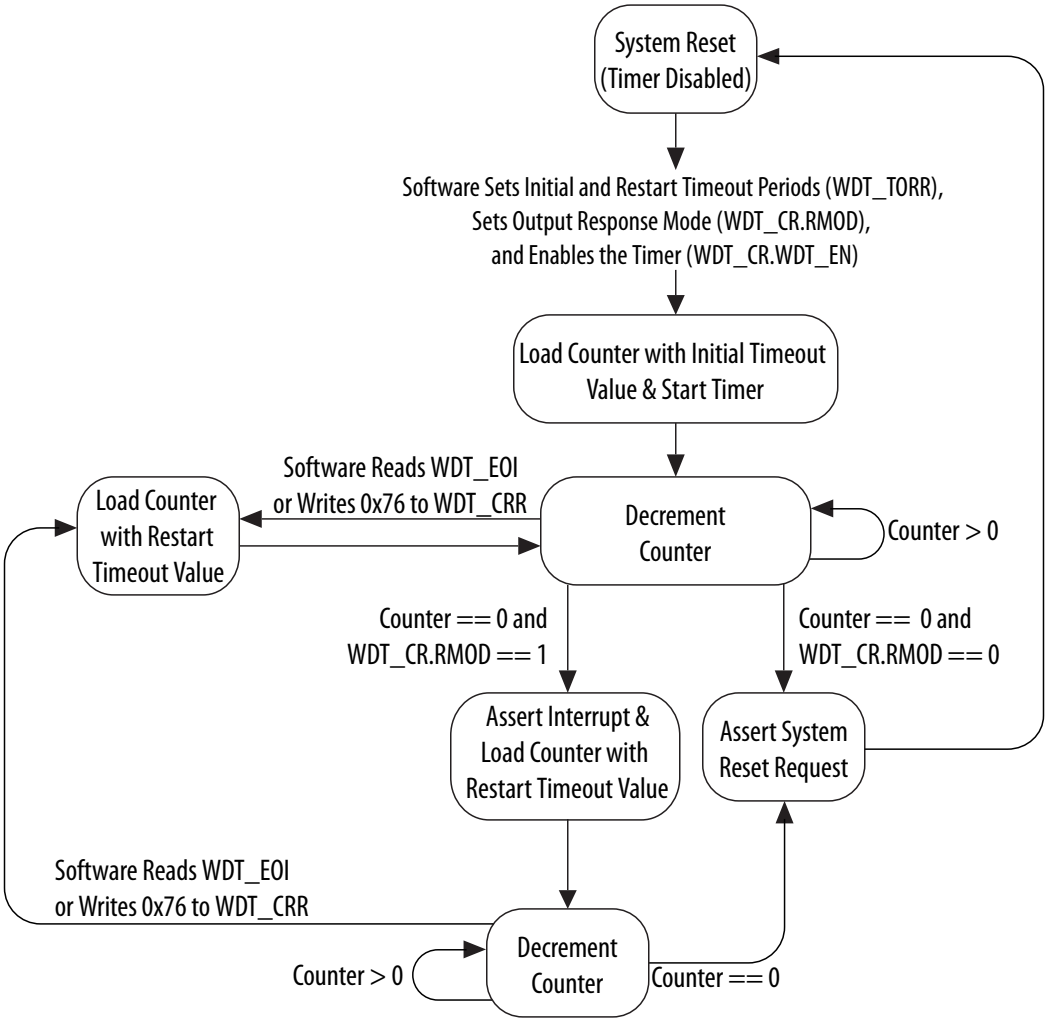
Related Information

Reset Manager on page 3-1
For more information, refer to the *Reset Manager* chapter.

Watchdog Timer State Machine

The following figure illustrates the behavior of the watchdog timer, including the behavior of both output response modes. Once initialized, the counter decrements at every clock cycle. The state machine remains in the Decrement Counter state until the counter reaches zero, or the watchdog timer is restarted. If software reads the interrupt clear register (`wdt_eoi`), or writes 0x76 to the `wdt_crr` register, the state changes from Decrement Counter to Load Counter with Restart Timeout Value. In this state, the watchdog counter gets reloaded with the restart timeout value, and then the state changes back to Decrement Counter.

Figure 24-2: Watchdog Timer State Machine



If the counter reaches zero, the state changes based on the value of the output response mode setting defined in the `rmod` bit of the `wdt_cr` register. If the `rmod` bit of the `wdt_cr` register is 0, the output response mode is to generate a system reset request. In this case, the state changes to Assert System Reset Request. In response, the reset manager resets and disables the watchdog timer, and gives software the opportunity to reinitialize the timer.

If the `rmod` bit of the `wdt_cr` register is 1, the output response mode is to generate an interrupt. In this case, the state changes to Assert Interrupt and Load Counter with Restart Timeout Value. An interrupt to the processor is generated, and the watchdog counter is reloaded with the restart timeout value. The state then changes to the second Decrement Counter state, and the counter resumes decrementing. If software reads the `wdt_eoi` register, or writes 0x76 to the `wdt_crr` register, the state changes from Decrement Counter to Load Counter with Restart Timeout Value. In this state, the watchdog counter gets reloaded with the restart timeout value, and then the state changes back to the first Decrement Counter state. If the counter again reaches zero, the state changes to Assert System Reset Request. In response, the reset manager resets the watchdog timer, and gives software the opportunity to reinitialize the timer.

Watchdog Timer Address Map and Register Definitions

For complete HPS address map and register definitions, refer to the **Arria 10 HPS Address Map and Register Definitions**.

watchdog Address Map

Module Instance	Base Address	End Address
i_watchdog_0_l4wd	0xFFD00200	0xFFD002FF
i_watchdog_1_l4wd	0xFFD00300	0xFFD01FFF

Important: To prevent indeterminate system behavior, reserved areas of memory must not be accessed by software or hardware. Any area of the memory map that is not explicitly defined as a register space or accessible memory is considered reserved.

Register	Offset	Width	Access	Reset Value	Description
wdt_cr on page 24-13	0x0	32	RW	0x0	Control Register
wdt_torr on page 24-15	0x4	32	RW	0xFF	Timeout Range Register
wdt_ccvr on page 24-17	0x8	32	RO	0x7FFFFFFF	Current Counter Value Register.
wdt_crr on page 24-18	0xC	32	WO	0x0	Counter Restart Register.
wdt_stat on page 24-19	0x10	32	RO	0x0	Interrupt Status Register.
wdt_eoi on page 24-20	0x14	32	RO	0x0	Interrupt Clear Register.
cp_wdt_user_top_max on page 24-21	0xE4	32	RO	0x0	Component Parameters Register 5
cp_wdt_user_top_init_max on page 24-21	0xE8	32	RO	0x0	Component Parameters Register 4
cd_wdt_top_rst on page 24-22	0xEC	32	RO	0xFF	Component Parameters Register 3
cp_wdt_cnt_rst on page 24-23	0xF0	32	RO	0x7FFFFFFF	Component Parameters Register 2

Register	Offset	Width	Access	Reset Value	Description
wdt_comp_param_1 on page 24-24	0xF4	32	RO	0x10FF0254	Component Parameters Register 1
wdt_comp_version on page 24-26	0xF8	32	RO	0x3130372A	Component Version Register
wdt_comp_type on page 24-27	0xFC	32	RO	0x44570120	Component Type Register

watchdog Summary

Module Instance	Base Address
i_watchdog_0_l4wd	0xFFD00200
i_watchdog_1_l4wd	0xFFD00300

Register Address Offset	Bit Fields															
i_watchdog_0_l4wd																
wdt_cr 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved											rpl RO 0x0			rmod RW 0x0	wdt_en RW 0x0
wdt_torr 0x4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved RO 0x0								top_init RW 0xF				top RW 0xF			
wdt_ccvr 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	wdt_ccvr RO 0x7FFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	wdt_ccvr RO 0x7FFFFFFF															

Register Address Offset	Bit Fields															
wdt_crr 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								wdt_crr WO 0x0							
wdt_stat 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															wdt_stat RO 0x0
wdt_eoi 0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															wdt_eoi RO 0x0
cp_wdt_user_top_max 0xE4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cp_wdt_user_top_max RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cp_wdt_user_top_max RO 0x0															
cp_wdt_user_top_init_max 0xE8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cp_wdt_user_top_init_max RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cp_wdt_user_top_init_max RO 0x0															
cd_wdt_top_rst 0xEC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cd_wdt_top_rst RO 0xFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cd_wdt_top_rst RO 0xFF															

Register Address Offset	Bit Fields															
cp_wdt_cnt_rst 0xF0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cp_wdt_cnt_rst RO 0x7FFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cp_wdt_cnt_rst RO 0x7FFFFFFF															
wdt_comp_param_1 0xF4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	rsvd_31_29 RO 0x0			cp_wdt_cnt_width RO 0x10					cp_wdt_dflt_top_init RO 0xF				cp_wdt_dflt_top RO 0xF			
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_15_13 RO 0x0			cp_wdt_dflt_rpl RO 0x0			cp_wdt_apb_data_width RO 0x2		cp_wdt_pause RO 0x0	cp_wdt_use_fix_top RO 0x1	cp_wdt_hc_top RO 0x0	cp_wdt_hc_rpl RO 0x1	cp_wdt_hc_rmod RO 0x0	cp_wdt_dual_top RO 0x1	cp_wdt_dflt_rmod RO 0x0	cp_wdt_always_en RO 0x0
wdt_comp_version 0xF8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	wdt_comp_version RO 0x3130372A															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	wdt_comp_version RO 0x3130372A															
wdt_comp_type 0xFC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	wdt_comp_type RO 0x44570120															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	wdt_comp_type RO 0x44570120															
i_watchdog_1_l4wd																
wdt_cr 0x0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved											rpl RW 0x0			rmod RW 0x0	wdt_en RW 0x0

Register Address Offset	Bit Fields															
wdt_torr 0x4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved RO 0x0								top_init RW 0xF				top RW 0xF			
wdt_ccvr 0x8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	wdt_ccvr RO 0x7FFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	wdt_ccvr RO 0x7FFFFFFF															
wdt_crr 0xC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								wdt_crr WO 0x0							
wdt_stat 0x10	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															wdt_stat RO 0x0
wdt_eoi 0x14	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	Reserved															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															wdt_eoi RO 0x0
cp_wdt_user_top_max 0xE4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cp_wdt_user_top_max RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cp_wdt_user_top_max RO 0x0															

Register Address Offset	Bit Fields															
cp_wdt_user_top_init_max 0xE8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cp_wdt_user_top_init_max RO 0x0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cp_wdt_user_top_init_max RO 0x0															
cd_wdt_top_rst 0xEC	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cd_wdt_top_rst RO 0xFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cd_wdt_top_rst RO 0xFF															
cp_wdt_cnt_rst 0xF0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	cp_wdt_cnt_rst RO 0x7FFFFFFF															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	cp_wdt_cnt_rst RO 0x7FFFFFFF															
wdt_comp_param_1 0xF4	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	rsvd_31_29 RO 0x0			cp_wdt_cnt_width RO 0x10					cp_wdt_dflt_top_init RO 0xF				cp_wdt_dflt_top RO 0xF			
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	rsvd_15_13 RO 0x0			cp_wdt_dflt_rpl RO 0x0			cp_wdt_apb_data_width RO 0x2		cp_wdt_pause RO 0x0	cp_wdt_use_fix_top RO 0x1	cp_wdt_hc_top RO 0x0	cp_wdt_hc_rpl RO 0x1	cp_wdt_hc_rmod RO 0x0	cp_wdt_dual_top RO 0x1	cp_wdt_dflt_rmod RO 0x0	cp_wdt_always_en RO 0x0
wdt_comp_version 0xF8	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	wdt_comp_version RO 0x3130372A															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	wdt_comp_version RO 0x3130372A															

Register Address Offset	Bit Fields															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
wdt_comp_type 0xFC	wdt_comp_type RO 0x44570120															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	wdt_comp_type RO 0x44570120															

wdt_cr

Control Register

Module Instance	Base Address	Register Address
i_watchdog_0_l4wd	0xFFD00200	0xFFD00200
i_watchdog_1_l4wd	0xFFD00300	0xFFD00300

Offset: 0x0

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved											rpl RO 0x0			rmod RW 0x0	wdt_en RW 0x0

wdt_cr Fields

Bit	Name	Description	Access	Reset
4:2	rpl	Reset pulse length.This field identifies the number of pclk cycles for which the system reset stays asserted. 000 - 2 pclk cycles 001 - 4 pclk cycles 010 - 8 pclk cycles 011 - 16 pclk cycles 100 - 32 pclk cycles 101 - 64 pclk cycles 110 - 128 pclk cycles 111 - 256 pclk cycles	RO	0x0
1	rmod	Response mode. Selects the output response generated to a timeout. 0 = Generate a system reset. 1 = First generate an interrupt and if it is not cleared by the time a second timeout occurs then generate a system reset. **Value** **Description** 0x0 RST 0x1 IRQRST	RW	0x0
0	wdt_en	WDT enable. Writable when the configuration parameter WDT_ALWAYS_EN = 0, otherwise, it is readable. This bit is used to enable and disable the DW_apb_wdt. When disabled, the counter does not decrement. Thus, no interrupts or system resets are generated. Once this bit has been enabled, it can be cleared only by a system reset. 0 = WDT disabled. 1 = WDT enabled. **Value** **Description** 0x0 DISABLED 0x1 ENABLED	RW	0x0

wdt_torr

Timeout Range Register

Module Instance	Base Address	Register Address
i_watchdog_0_l4wd	0xFFD00200	0xFFD00204
i_watchdog_1_l4wd	0xFFD00300	0xFFD00304

Offset: 0x4

Access: RW

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
reserved RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved RO 0x0								top_init RW 0xF				top RW 0xF			

wdt_torr Fields

Bit	Name	Description	Access	Reset
31:8	reserved	Reserved and read as 0.	RO	0x0
7:4	top_init	Timeout period for initialization. Used to select the timeout period that the watchdog counter restarts from for the first counter restart (kick) . This register should be written after reset and before the WDT is enabled. A change of the timeout period for initialization (TOP_INIT) is seen only once the WDT has been enabled, and any change after the first kick is not seen as subsequent kicks use the period specified by the TOP bits. If TOP_INIT is programmed to select a range that is greater than the counter width of 32, the timeout period is truncated to fit to the counter width. This	RW	0xF

Bit	Name	Description	Access	Reset
		affects only the non-user specified values as users are limited to these boundaries during configuration. **Value** **Description** 0xa TIMEOUT64M 0xb TIMEOUT128M 0xc TIMEOUT256M 0xd TIMEOUT512M 0xe TIMEOUT1G 0xf TIMEOUT2G 0x0 TIMEOUT64K 0x1 TIMEOUT128K 0x2 TIMEOUT256K 0x3 TIMEOUT512K 0x4 TIMEOUT1M 0x5 TIMEOUT2M 0x6 TIMEOUT4M 0x7 TIMEOUT8M 0x8 TIMEOUT16M 0x9 TIMEOUT32M		
3:0	top	Timeout period. This field is used to select the timeout period from which the watchdog counter restarts. A change of the timeout period takes effect only after the next counter restart (kick). If the time out period (TOP) is programmed to select a range that is greater than the counter width of 32, the timeout period is truncated to fit to the counter width. This affects only the non-user specified values as users are limited to these boundaries during configuration. **Value** **Description** 0xa TIMEOUT64M 0xb TIMEOUT128M 0xc TIMEOUT256M 0xd TIMEOUT512M 0xe TIMEOUT1G 0xf TIMEOUT2G	RW	0xF

Bit	Name	Description	Access	Reset
		Value: Description 0x0: TIMEOUT64K 0x1: TIMEOUT128K 0x2: TIMEOUT256K 0x3: TIMEOUT512K 0x4: TIMEOUT1M 0x5: TIMEOUT2M 0x6: TIMEOUT4M 0x7: TIMEOUT8M 0x8: TIMEOUT16M 0x9: TIMEOUT32M		

wdt_ccvr

Current Counter Value Register.

Module Instance	Base Address	Register Address
i_watchdog_0_l4wd	0xFFD00200	0xFFD00208
i_watchdog_1_l4wd	0xFFD00300	0xFFD00308

Offset: 0x8

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
wdt_ccvr RO 0x7FFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
wdt_ccvr RO 0x7FFFFFFF															

wdt_ccvr Fields

Bit	Name	Description	Access	Reset
31:0	wdt_ccvr	This register, when read, is the current value of the internal counter. This value is read coherently when ever it is read.	RO	0x7FFFFFFF

wdt_crr

Counter Restart Register.

Module Instance	Base Address	Register Address
i_watchdog_0_l4wd	0xFFD00200	0xFFD0020C
i_watchdog_1_l4wd	0xFFD00300	0xFFD0030C

Offset: 0xC

Access: WO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								wdt_crr WO 0x0							

wdt_crr Fields

Bit	Name	Description	Access	Reset
7:0	wdt_crr	This register is used to restart the WDT counter. As a safety feature to prevent accidental restarts, the value 0x76 must be written. A restart also clears the WDT interrupt. Reading this register returns zero. **Value** **Description** 0x76 KICK	WO	0x0

wdt_stat

Interrupt Status Register.

Module Instance	Base Address	Register Address
i_watchdog_0_l4wd	0xFFD00200	0xFFD00210
i_watchdog_1_l4wd	0xFFD00300	0xFFD00310

Offset: 0x10

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															wdt_stat RO 0x0

wdt_stat Fields

Bit	Name	Description	Access	Reset
0	wdt_stat	This register shows the interrupt status of the WDT. 1 = Interrupt is active regardless of polarity. 0 = Interrupt is inactive. **Value** — **Description** 0x0 — INACTIVE 0x1 — ACTIVE	RO	0x0

wdt_eoi

Interrupt Clear Register.

Module Instance	Base Address	Register Address
i_watchdog_0_l4wd	0xFFD00200	0xFFD00214
i_watchdog_1_l4wd	0xFFD00300	0xFFD00314

Offset: 0x14

Access: RO

Important: The value of a reserved bit must be maintained in software. When you modify registers containing reserved bit fields, you must use a read-modify-write operation to preserve state and prevent indeterminate system behavior.

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															wdt_eoi RO 0x0

wdt_eoi Fields

Bit	Name	Description	Access	Reset
0	wdt_eoi	Clears the watchdog interrupt. This can be used to clear the interrupt without restarting the watchdog counter.	RO	0x0

cp_wdt_user_top_max

Component Parameters Register 5

Module Instance	Base Address	Register Address
i_watchdog_0_l4wd	0xFFD00200	0xFFD002E4
i_watchdog_1_l4wd	0xFFD00300	0xFFD003E4

Offset: 0xE4

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cp_wdt_user_top_max RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cp_wdt_user_top_max RO 0x0															

cp_wdt_user_top_max Fields

Bit	Name	Description	Access	Reset
31:0	cp_wdt_user_top_max	Upper limit of Timeout Period parameters.	RO	0x0

cp_wdt_user_top_init_max

Component Parameters Register 4

Module Instance	Base Address	Register Address
i_watchdog_0_l4wd	0xFFD00200	0xFFD002E8
i_watchdog_1_l4wd	0xFFD00300	0xFFD003E8

Offset: 0xE8

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cp_wdt_user_top_init_max RO 0x0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cp_wdt_user_top_init_max RO 0x0															

cp_wdt_user_top_init_max Fields

Bit	Name	Description	Access	Reset
31:0	cp_wdt_user_top_init_max	Upper limit of Initial Timeout Period parameters.	RO	0x0

cd_wdt_top_rst

Component Parameters Register 3

Module Instance	Base Address	Register Address
i_watchdog_0_l4wd	0xFFD00200	0xFFD002EC
i_watchdog_1_l4wd	0xFFD00300	0xFFD003EC

Offset: 0xEC

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cd_wdt_top_rst RO 0xFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cd_wdt_top_rst RO 0xFF															

cd_wdt_top_rst Fields

Bit	Name	Description	Access	Reset
31:0	cd_wdt_top_rst	Contains the reset value of the WDT_TORR register.	RO	0xFF

cp_wdt_cnt_rst

Component Parameters Register 2

Module Instance	Base Address	Register Address
i_watchdog_0_l4wd	0xFFD00200	0xFFD002F0
i_watchdog_1_l4wd	0xFFD00300	0xFFD003F0

Offset: 0xF0

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cp_wdt_cnt_rst RO 0x7FFFFFFF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cp_wdt_cnt_rst RO 0x7FFFFFFF															

cp_wdt_cnt_rst Fields

Bit	Name	Description	Access	Reset
31:0	cp_wdt_cnt_rst	The timeout period range is fixed. The range increments by the power of 2 from 2 to the 16 to 2 to the 31.	RO	0x7FFFFFFF

wdt_comp_param_1

Component Parameters Register 1

Module Instance	Base Address	Register Address
i_watchdog_0_l4wd	0xFFD00200	0xFFD002F4
i_watchdog_1_l4wd	0xFFD00300	0xFFD003F4

Offset: 0xF4

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
rsvd_31_29 RO 0x0			cp_wdt_cnt_width RO 0x10					cp_wdt_dflt_top_init RO 0xF				cp_wdt_dflt_top RO 0xF			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
rsvd_15_13 RO 0x0			cp_wdt_dflt_rpl RO 0x0			cp_wdt_apb_data_width RO 0x2		cp_wdt_pause RO 0x0	cp_wdt_use_fix_top RO 0x1	cp_wdt_hc_top RO 0x0	cp_wdt_hc_rpl RO 0x1	cp_wdt_hc_rmod RO 0x0	cp_wdt_dual_top RO 0x1	cp_wdt_dflt_rmod RO 0x0	cp_wdt_always_en RO 0x0

wdt_comp_param_1 Fields

Bit	Name	Description	Access	Reset
31:29	rsvd_31_29		RO	0x0
28:24	cp_wdt_cnt_width	Value: 0x10 — Description: WIDTH32BITS	RO	0x10

Bit	Name	Description	Access	Reset
23:20	cp_wdt_dflt_top_init	Value Description 0xf TIMEOUT15	RO	0xF
19:16	cp_wdt_dflt_top	Value Description 0xf TIMEOUT15	RO	0xF
15:13	rsvd_15_13		RO	0x0
12:10	cp_wdt_dflt_rpl	Value Description 0x0 PULSE2CYCLES	RO	0x0
9:8	cp_wdt_apb_data_width	Value Description 0x2 WIDTH32BITS	RO	0x2
7	cp_wdt_pause		RO	0x0
6	cp_wdt_use_fix_top	Value Description 0x1 PREDEFINED	RO	0x1
5	cp_wdt_hc_top	Value Description 0x0 PROGRAMMABLE	RO	0x0
4	cp_wdt_hc_rpl	Value Description 0x1 HARDCODED	RO	0x1
3	cp_wdt_hc_rmod	Value Description 0x0 PROGRAMMABLE	RO	0x0
2	cp_wdt_dual_top	Value Description 0x1 DUALTOP	RO	0x1

Bit	Name	Description	Access	Reset
1	cp_wdt_dflt_rmod	Value: 0x0 — Description: RSTREQ	RO	0x0
0	cp_wdt_always_en	Value: 0x0 — Description: DISABLED	RO	0x0

wdt_comp_version

Component Version Register

Module Instance	Base Address	Register Address
i_watchdog_0_l4wd	0xFFD00200	0xFFD002F8
i_watchdog_1_l4wd	0xFFD00300	0xFFD003F8

Offset: 0xF8

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
wdt_comp_version RO 0x3130372A															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
wdt_comp_version RO 0x3130372A															

wdt_comp_version Fields

Bit	Name	Description	Access	Reset
31:0	wdt_comp_version	ASCII value for each number in the version, followed by *. For example, 32_30_31_2A represents the version 2.01*.	RO	0x3130372A

wdt_comp_type

Component Type Register

Module Instance	Base Address	Register Address
i_watchdog_0_l4wd	0xFFD00200	0xFFD002FC
i_watchdog_1_l4wd	0xFFD00300	0xFFD003FC

Offset: 0xFC

Access: RO

Bit Fields															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
wdt_comp_type RO 0x44570120															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
wdt_comp_type RO 0x44570120															

wdt_comp_type Fields

Bit	Name	Description	Access	Reset
31:0	wdt_comp_type	Component Type Register	RO	0x44570120

Document Revision History

Table 24-3: Document Revision History

Date	Version	Changes
October 2016	2016.10.28	Maintenance release.
May 2016	2016.05.27	Maintenance release.
May 2016	2016.05.03	Maintenance release.
November 2015	2015.11.02	Added note to "Watchdog Timer Counter" section.
May 2015	2015.05.04	Maintenance release.

Date	Version	Changes
December 2014	2014.12.15	• Maintenance release. • Added *Taking the Watchdog Timer Out of Reset* section.
August 2014	2014.08.18	Initial release

Hard Processor System I/O Pin Multiplexing 25

2016.10.28

a10_5v4 Subscribe Send Feedback

The Arria 10 SoC has two banks of flexible I/O pins that are used for hard processor system (HPS) operation, external memory and external peripheral connection. Some of the pins are dedicated to the HPS, and others can be shared with the FPGA fabric. A pin multiplexing mechanism allows the SoC to use the flexible I/O pins in a wide range of configurations.

Features of the HPS I/O Block

The I/O block provides the following functionality and features:

- I/O pins
  - Dedicated I/O – 17 pins supporting clock, resets, boot devices, and other key peripherals.
  - Shared I/O – 48 pins available for HPS external memory and peripherals. Shared I/O pins can also be used by FPGA logic.

  Note: The HPS also interfaces with an SDRAM memory controller. This interface is separate from the dedicated and shared I/O pins discussed in this chapter.
- I/O multiplexing
  - Selects pins used by each HPS peripheral
  - Can assign shared I/O pins to FPGA logic
  - Can expose HPS peripheral interfaces to FPGA logic

  Note: When routed to the FPGA, some HPS peripherals require additional pipeline support in the connected soft logic. Refer to the relevant HPS peripheral chapter for details.

  I/O multiplexing is configured when you instantiate the HPS component.

Related Information

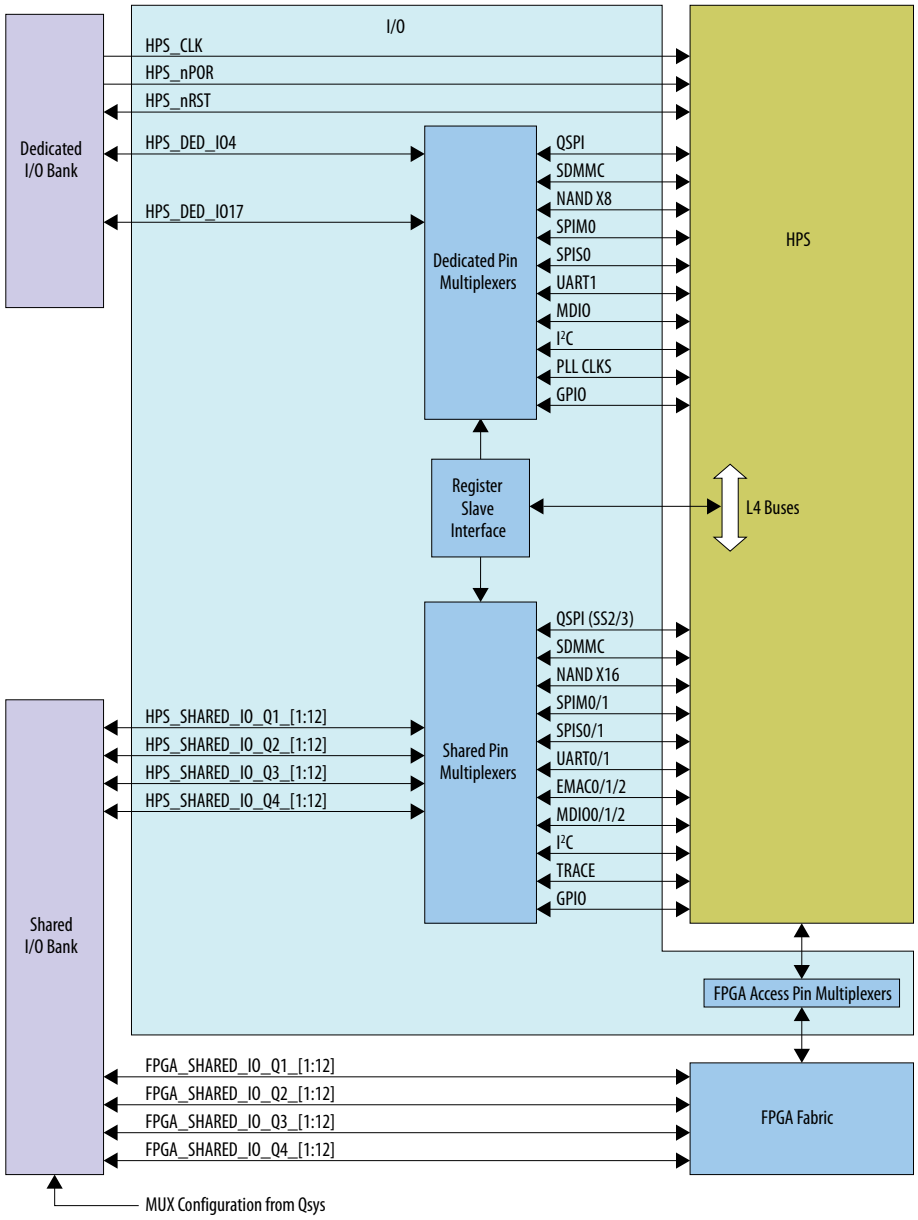
- **External Memory Interface Handbook**
  For details about memory I/O pins in the SoC hard memory controller, refer to the *Functional Description - HPS Memory Controller* chapter of the *External Memory Interface Handbook*.
- **Configuring Peripheral Pin Multiplexing** on page 27-14
  For information about configuring the HPS I/O MUXes during system configuration, refer to the *Configuring Peripheral Pin Multiplexing* chapter.

HPS I/O Block Diagram and System Integration

The HPS I/O block consists of the following subblocks:

- Dedicated pin multiplexers (MUXes) – MUXes for the dedicated I/O bank
- Shared pin multiplexers – MUXes for the shared I/O bank
- FPGA access pin multiplexers – MUXes for HPS peripheral connections to the FPGA fabric
- Register slave interface – Provides access to control registers, which allow the bootloader to initialize I/O pins and HPS peripheral interfaces at system startup

Figure 25-1: HPS I/O Block Diagram



Related Information

Control Registers on page 25-5

Functional Description of the HPS I/O

Dedicated I/O Pins

The HPS has 17 dedicated I/O pins. Three are used as clock, cold reset and warm reset. The remaining 14 dedicated I/O pins are dedicated to boot devices and other commonly-used peripherals. The following HPS peripherals can be served by dedicated I/O pins:

- GPIO2
- NAND
- SD/MMC
- QSPI
- CM_PLL
- UART1
- SPI0
- EMAC0 MDIO interface
- EMAC1 MDIO interface
- EMAC2 MDIO interface

Each of these peripherals can alternatively be routed through the FPGA. You can configure this routing when you instantiate the HPS component.

Related Information

- **Booting and Configuration** on page 30-1
  Details about boot select pins
- **FPGA Access** on page 25-5
  Information about routing HPS peripheral interfaces to the FPGA
- **Configuring Peripheral Pin Multiplexing** on page 27-14
  For information about configuring the HPS I/O MUXes during system configuration, refer to the *Configuring Peripheral Pin Multiplexing* chapter.
- **Configuring HPS I/O Multiplexing** on page 25-8
  Information about configuring the HPS I/O MUXes

Warm Reset Pin

The warm reset signal is a bidirectional signal; a warm reset event can drive into the HPS, or a warm reset event can be generated by the HPS and drive out of this pin.

Boot Pins

The following dedicated I/O pins convey boot source information:

- `HPS_DEDICATED_6/BOOTSEL2`
- `HPS_DEDICATED_10/BOOTSEL1`
- `HPS_DEDICATED_11/BOOTSEL0`

Typically, software samples these pins at cold reset.

Related Information

- **Arria 10 Hard Processor System Pin Information spreadsheet**
  Details of available HPS pin MUXing options
- **Pin-Out Files for Altera Devices**
  Pinout files for Arria 10 devices

Shared I/O Pins

There are 48 HPS peripheral pins that are shared with the FPGA core. They are divided into four quadrants of 12 signals per quadrant. Each quadrant can be assigned to the HPS or the FPGA fabric.

In each shared I/O quadrant, all 12 I/Os are assigned either to the FPGA or to the HPS. It is not possible to divide the I/Os in a quadrant between the FPGA and HPS.

When a quadrant is assigned to the FPGA fabric, you can connect FPGA soft logic to the shared I/O pins with the assignment editor, just as for any other FPGA I/O pins. Soft logic can use the shared I/O pins to interface to off-chip resources. However, pins in that quadrant are no longer directly available to HPS peripherals. Enabled HPS peripherals can be assigned to another quadrant, or else their interfaces can be routed to the FPGA fabric.

The following HPS peripherals can be assigned to shared I/O pins:

- EMAC0
- EMAC1
- EMAC2
- GPIO0
- GPIO1
- I2C0
- I2C1
- NAND
- QSPI
- SD/MMC
- SPI0
- SPI1
- TRACE
- UART0
- UART1
- USB0
- USB1

Each of these peripherals except USB0 and USB1 can alternatively be routed through the FPGA. You configure this routing when you instantiate the HPS component in Qsys.

Note: Although the shared I/O pins are configured through the control registers, software cannot reconfigure the shared I/O pins after I/O configuration is complete. There is no support for dynamically changing the pin MUX selections for shared pins.

Related Information

- **FPGA Access** on page 25-5
  Information about routing HPS peripheral interfaces to the FPGA

- **Configuring Peripheral Pin Multiplexing** on page 27-14
  For information about configuring the HPS I/O MUXes during system configuration, refer to the *Configuring Peripheral Pin Multiplexing* chapter.
- **HPS I/O Block Diagram and System Integration** on page 25-2
  Refer to the HPS I/O block diagram for an overview of how the HPS and FPGA share the shared I/O bank.
- **Configuring HPS I/O Multiplexing** on page 25-8
  Information about configuring the HPS I/O MUXes

FPGA Access

Most HPS peripheral interfaces can be connected into the FPGA fabric, instead of to the dedicated or shared I/O pins.

HPS peripherals connect to the FPGA fabric through the FPGA access pin MUX. When connected to the FPGA fabric, peripheral interfaces are exposed as ports of the HPS component.

Connecting HPS peripherals to the FPGA fabric can be a strategy to make optimal use of the I/O pins available to the HPS. For example, you can route HPS peripherals through the FPGA if your design requires more I/Os than the HPS I/O block provides.

All HPS peripherals except the USB 2.0 OTG controllers can interface to the FPGA fabric.

Related Information

Configuring HPS I/O Multiplexing on page 25-8
Information about configuring the HPS I/O MUXes

Control Registers

The HPS provides control registers that allow the system to initialize the following I/O parameters at system startup:

- Pins assigned to each HPS peripheral
- Shared I/O pins optionally assigned to FPGA logic
- HPS peripheral interfaces optionally exposed to FPGA logic
- I/O cell configuration

The HPS I/O control registers can only be accessed in secure mode.

Control registers can be divided into the following groups:

- Dedicated pin MUX registers
- Dedicated configuration registers
- Shared pin MUX registers
- FPGA access MUX registers

You set up the control registers when you instantiate the HPS component at the time of system generation. When you configure the HPS component, Qsys determines the correct register settings, and places them in the Quartus® Prime I/O programming file.

Related Information

- **http://www.altera.com/literature/hb/arria-10/hps.html**
- **Configuring HPS I/O Multiplexing** on page 25-8
  Information about configuring the HPS I/O MUXes

Dedicated Pin MUX Registers

Dedicated pin MUX registers control the functions of the dedicated pins. One register is provided for each dedicated I/O pin.

The HPS provides pin MUX registers, `pinmux_dedicated_io_4` through `pinmux_dedicated_io_17`, for each of the dedicated pins HPS_DEDICATED_4 through HPS_DEDICATED_17. Each pin MUX register contains a 4-bit MUX select field to select the function of the dedicated pin. At a cold reset event these fields are reset to 15, selecting a general purpose I/O function. A warm reset event does not affect these registers.

The registers for dedicated I/O pins HPS_DEDICATED_1 through HPS_DEDICATED_3 provide no functionality, because these pins are always used as the HPS clock and reset pins.

Note: Although the dedicated I/O pins are configured through the control registers, Altera recommends against reconfiguring the dedicated I/O pins after I/O configuration is complete.

Related Information

- **http://www.altera.com/literature/hb/arria-10/hps.html**
- **Arria 10 Hard Processor System Pin Information spreadsheet**
  Details of available HPS pin MUXing options
- **Pin-Out Files for Altera Devices**
  Pinout files for Arria 10 devices

Dedicated Configuration Registers

Configuration registers for each dedicated I/O pin allow software to control the corresponding I/O cell. These registers, `configuration_dedicated_io_1` through `configuration_dedicated_io_17`, allow software to set the following characteristics:

- Termination
- Slew rate
- Input buffer settings

These registers are not affected by a warm reset.

One configuration register, `configuration_dedicated_io_bank`, is also provided to select the I/O voltage for the dedicated I/O bank. At cold reset this register defaults to 0x2 (2.5-3.0 volt operation). This register is not affected by a warm reset.

Typically, the boot code programs this register when the system boots up. The boot code reads the BOOTSEL0, BOOTSEL1, and BOOTSEL2 pins and then sets the dedicated I/O bank to either 1.8 or 2.5-3.0 volt operation, depending on the pin status.

Note: Although the dedicated I/O pins are configured through the control registers, Altera recommends against reconfiguring the dedicated I/O pins after I/O configuration is complete.

Related Information

- **http://www.altera.com/literature/hb/arria-10/hps.html**
- **Configuring HPS I/O Multiplexing** on page 25-8
  Information about configuring the HPS I/O MUXes

Shared Pin MUX Registers

There are 48 shared pin MUX registers, one for each shared I/O pin. The pin MUX register names correspond to the signal names, as listed in the following table.

Table 25-1: Shared Pin MUX Register Names and Signals

Shared Pin MUX Register Name	HPS Signal Name	FPGA Signal Name
pinmux_shared_io_q1_1	HPS_SHARED_IO_Q1_1	FPGA_SHARED_IO_Q1_1
...	...	...
pinmux_shared_io_q1_12	HPS_SHARED_IO_Q1_12	FPGA_SHARED_IO_Q1_12
pinmux_shared_io_q2_1	HPS_SHARED_IO_Q2_1	FPGA_SHARED_IO_Q2_1
...	...	...
pinmux_shared_io_q2_12	HPS_SHARED_IO_Q2_12	FPGA_SHARED_IO_Q2_12
pinmux_shared_io_q3_1	HPS_SHARED_IO_Q3_1	FPGA_SHARED_IO_Q3_1
...	...	...
pinmux_shared_io_q3_12	HPS_SHARED_IO_Q3_12	FPGA_SHARED_IO_Q3_12
pinmux_shared_io_q4_1	HPS_SHARED_IO_Q4_1	FPGA_SHARED_IO_Q4_1
...	...	...
pinmux_shared_io_q4_12	HPS_SHARED_IO_Q4_12	FPGA_SHARED_IO_Q4_12

Each shared pin MUX register contains a 4-bit MUX select field to select the function of the shared I/O pin. At a cold reset event these fields default to 15, selecting a general purpose I/O function. A warm reset event does not affect these registers.

Note: Although the shared I/O pins are configured through the control registers, software cannot reconfigure the shared I/O pins after I/O configuration is complete. There is no support for dynamically changing the pin MUX selections for shared pins.

Related Information

- **http://www.altera.com/literature/hb/arria-10/hps.html**
- **Arria 10 Hard Processor System Pin Information spreadsheet**
  Details of available HPS pin MUXing options
- **Pin-Out Files for Altera Devices**
  Pinout files for Arria 10 devices

FPGA Access MUX Registers

The FPGA access ("use FPGA") MUX registers select whether each HPS peripheral uses HPS I/O pins or is routed to the FPGA fabric.

All peripherals except the USBs can be routed to the FPGA. The following FPGA access registers are available:

- `pinmux_emac0_usefpga`
- `pinmux_emac1_usefpga`
- `pinmux_emac2_usefpga`
- `pinmux_i2c0_usefpga`
- `pinmux_i2c1_usefpga`
- `pinmux_i2c_emac0_usefpga`
- `pinmux_i2c_emac1_usefpga`
- `pinmux_i2c_emac2_usefpga`
- `pinmux_nand_usefpga`
- `pinmux_qspi_usefpga`
- `pinmux_sdmmc_usefpga`
- `pinmux_spim0_usefpga`
- `pinmux_spim1_usefpga`
- `pinmux_spis0_usefpga`
- `pinmux_spis1_usefpga`
- `pinmux_uart0_usefpga`
- `pinmux_uart1_usefpga`
- `pinmux_mdio0_usefpga`
- `pinmux_mdio1_usefpga`
- `pinmux_mdio2_usefpga`

At cold reset, the FPGA access registers default to 0, selecting the HPS I/O pins. These registers are not affected by a warm reset event.

Note: Although the FPGA access MUX is configured through the control registers, Altera recommends against reconfiguring the FPGA access MUX after I/O configuration is complete.

Related Information

- **http://www.altera.com/literature/hb/arria-10/hps.html**
- **Configuring HPS I/O Multiplexing** on page 25-8
  Information about configuring the HPS I/O MUXes

Configuring HPS I/O Multiplexing

HPS I/O multiplexing is configured when the system is generated with Qsys.

Related Information

Configuring Peripheral Pin Multiplexing on page 27-14
For information about configuring the HPS I/O MUXes during system configuration, refer to the *Configuring Peripheral Pin Multiplexing* chapter.

Configuring Multiplexing at System Generation

There are 48 HPS peripheral pins that are shared with the FPGA core. They are divided into four quadrants of 12 signals per quadrant. Each quadrant can be configured with the Qsys system integration tool to be assigned to the HPS or the FPGA fabric.

The select for this multiplexer is in the I/O configuration shift registers (IOCSRs) which are configured by the Quartus Prime I/O programming file.

When you configure the HPS component, Qsys determines the correct register settings, and creates a device tree for the boot loader. When the system boots up, the boot loader configures the registers before configuring the I/O chain.

Configuring Multiplexing at Boot Time

Typically, the boot code programs the `configuration_dedicated_io_bank` register when the system boots up. The boot code reads the BOOTSEL0, BOOTSEL1, and BOOTSEL2 pins and then sets the dedicated I/O bank to either 1.8 V or 2.5-3.0 V operation, depending on the pin status.

Related Information

- **http://www.altera.com/literature/hb/arria-10/hps.html**
- **Control Registers** on page 25-5

Test Considerations

The dedicated I/O pins are chained into the full chip JTAG boundary scan chain. However, in some power modes the HPS can be off or disabled. The SoC device provides a bypass MUX to remove the dedicated I/O pins from the scan chain in this situation.

The boundary scan phase is not required to include I/O configuration shift register (IOCSR) configuration for the CONFIG_IO instruction. The only requirement is that no software be executing on the HPS during boundary scan.

If CONFIG_IO mode is active during the boundary scan phase, the HPS is in cold reset, preventing any software interference with the boundary scan. However, if the I/Os are configured in user mode, CONFIG_IO mode is not active during the scan phase, and you must implement safeguards to prevent software from executing during scan.

Related Information

Arria 10 Core Fabric and General Purpose I/Os Handbook
For information about JTAG boundary-scan testing in Arria 10 devices, refer to "JTAG Boundary-Scan Testing" in the *Arria 10 Devices* chapter of the *Arria 10 Core Fabric and General Purpose I/O Handbook*.

I/O Pin MUX Address Map and Register Definitions for Arria 10

For complete HPS address map and register definitions, refer to the **Arria 10 HPS Address Map and Register Definitions**.

Document Revision History

Table 25-2: Document Revision History

Date	Version	Changes
October 2016	2016.10.28	Maintenance release
May 2016	2016.05.27	Maintenance release.
May 2016	2016.05.03	Maintenance release
November 2015	2015.11.02	Maintenance release
May 2015	2015.05.04	Added address maps and register definitions
August 2014	2014.08.18	Initial release

Introduction to the HPS Component 26

2016.10.28

a10_5v4 Subscribe Send Feedback

The hard processor system (HPS) component is a wrapper that interfaces logic in the user design to the HPS hard logic, simulation models, BFMs, and software handoff files. It instantiates the HPS hard logic in the user design; and enables other soft components to interface with the HPS hard logic. The HPS component itself has a small footprint in the FPGA fabric, because its only purpose is to enable soft logic to connect to the extensive hard logic in the HPS. You can connect soft logic to the HPS.

After the soft logic is connected to the HPS, Qsys ensures the following features:

- Interoperability by adapting Avalon® Memory-Mapped (Avalon-MM) interfaces to AXI
- Handle data width mismatches
- Clock domain transfer crossing

This allows you to integrate IP from Altera, 3rd party IP cores, and custom IP cores to the HPS without having to create integration logic.

For a description of the HPS and its integration into the system on a chip (SoC), refer to the *Arria 10 Device Datasheet*.

For a description of the HPS system architecture and features, refer to the "Introduction to the Hard Processor" and the CoreSight Debug and Trace chapters in the *Arria 10 Device Handbook*.

For more information about instantiating the HPS component, refer to the *Instantiating the HPS Component* chapter in the *Hard Processor System Technical Reference Manual*.

For more information about the HPS component interfaces, refer to the *HPS Component Interfaces* chapter in the *Hard Processor System Technical Reference Manual*.

For more information about simulating the HPS component, refer to the *Simulating the HPS Component* chapter in the *Hard Processor System Technical Reference Manual*.

Related Information

- **Arria 10 Device Datasheet**
- **Simulating the HPS Component** on page 29-1
- **HPS Component Interfaces** on page 28-1
- **Instantiating the HPS Component** on page 27-1
- **Introduction to the Hard Processor System** on page 1-1
- **CoreSight Debug and Trace** on page 10-1

MPU Subsystem

The MPU subsystem features the dual ARM Cortex-A9 MPCore processors.

Related Information

Cortex-A9 Microprocessor Unit Subsystem on page 9-1

ARM CoreSight Debug Components

The following lists the ARM CoreSight debug components:

- Debug Access Port (DAP)
- System Trace Macrocell (STM)
- Trace Funnel
- Embedded Trace FIFO (ETF)
- AMBA Trace Bus Replicator (Replicator)
- Embedded Trace Router (ETR)
- Trace Port Interface Unit (TPIU)
- Embedded Cross Trigger (ECT)
- Program Trace Macrocell (PTM)

Related Information

CoreSight Debug and Trace on page 10-1

Interconnect

The interconnect consists of the L3 interconnect, SDRAM L3 interconnect, and level 4 (L4) buses. The L3 interconnect is an Arteris FlexNoC network-on-chip (NOC) interconnect module.

Related Information

System Interconnect on page 7-1

HPS-to-FPGA Interfaces

The HPS-to-FPGA interfaces provide a variety of communication channels between the HPS and the FPGA fabric. The HPS is highly integrated with the FPGA fabric, resulting in thousands of connecting signals. Some of the HPS-to-FPGA interfaces include:

- FPGA-to-HPS port
- HPS-to-FPGA port
- Lightweight HPS-to-FPGA port
- FPGA-to-SDRAM interface

Related Information

HPS-FPGA Bridges on page 8-1

Memory Controllers

The following lists the memory controller peripherals:

- SDRAM L3 Interconnect
- NAND Flash Controller
- Quad SPI Controller
- SD/MMC Controller

Related Information

- **System Interconnect** on page 7-1
- **NAND Flash Controller** on page 13-1
- **SD/MMC Controller** on page 14-1
- **Quad SPI Flash Controller** on page 15-1

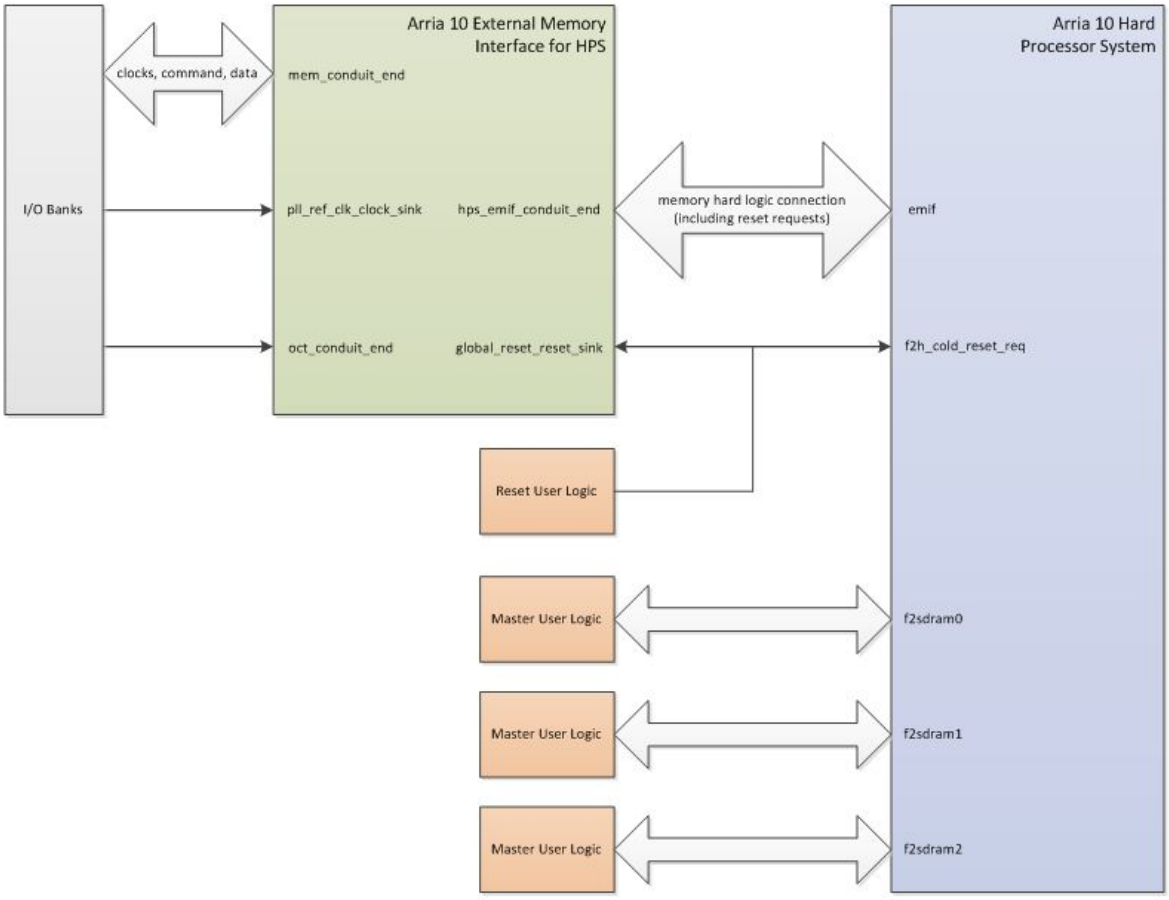
HPS SDRAM Controller

There are several memory interfaces connected, but only one connection to each interface is allowed at one time. You must instantiate a separate SDRAM controller and connect it to the HPS.

HMC and HPS Connectivity

The following figure shows that the HMC clock comes from the I/O itself, the HPS can reset the memory through the dedicated hard-logic connection between them, and any logic in the FPGA that resets the HMC must also be connected to the HPS.

Figure 26-1: HMC and HPS Connectivity



Clocks

You must connect the SDRAM controller to a reference clock that comes from the I/O column where the RAM controller is located.

Resets

There are two ways that the SDRAM controller can be connected to reset.

- One way - Reset and status connection is automatic by just connecting the SDRAM Hard Memory Controller (HMC) to the HPS. You access these wires in the reset manager so that you can reset the HMC and find out when the reset is complete.
- Second way - The HMC reset input from the FPGA fabric. This signal is called `global_reset_n` and is one of the signals exposed to the FPGA fabric from the HMC when you instantiate it in Qsys. The purpose of this signal is to reset the HMC from the FPGA side of the fabric. Before the FPGA fabric is configured, the memory controller ignores this signal which is why the memory controller can become live before the fabric is configured. This also means that the user design that gets configured into the fabric must drive it because once the fabric is configured, the HMC will use `global_reset_n` as an input source.

Note: You must connect the input source for `global_reset_n` to the cold reset input for the HPS. If you put the SDRAM into reset but that same reset source does not connect to the HPS, the memory is then wiped out while the HPS is still trying to access it. This causes the software to crash.

Related Information

Functional Description - HPS Memory Controller

Support Peripherals

The following lists the support peripherals:

- Clock Manager
- Reset Manager
- Security Manager
- System Timers
- System Watchdog Timers
- Direct Memory Access (DMA) Controller
- FPGA Manager

Related Information

- **Clock Manager** on page 2-1
- **Reset Manager** on page 3-1
- **FPGA Manager** on page 4-1
- **System Manager** on page 5-1
- **SoC Security** on page 6-1
  For more information, refer to the *Security Manager* chapter in the *Hard Processor System Technical Reference Manual*.
- **DMA Controller** on page 16-1
- **Timer** on page 23-1
- **Watchdog Timer** on page 24-1

Interface Peripherals

The following lists the interface peripherals:

- Ethernet Media Access Controller
- USB 2.0 On-The-Go (OTG) Controllers
- I^2C Controllers
- UART
- SPI Master Controllers
- SPI Slave Controllers
- GPIO Interfaces

Related Information

- **Ethernet Media Access Controller** on page 17-1
- **USB 2.0 OTG Controller** on page 18-1
- **SPI Controller** on page 19-1

- **I2C Controller** on page 20-1
- **UART Controller** on page 21-1
- **General-Purpose I/O Interface** on page 22-1

On-Chip Memories

The following lists the on-chip memories:

- On-Chip RAM
- Boot ROM

Related Information

On-Chip Memory on page 12-1

Document Revision History

Table 26-1: Document Revision History

Date	Version	Changes
October 2016	2016.10.28	Maintenance release
May 2016	2016.05.27	Maintenance Release
May 2016	2016.05.03	Removed FPGA-to-HPS SDRAM interface
November 2015	2015.11.02	Maintenance release
May 2015	2015.05.04	Maintenance release
December 2014	2014.12.15	Maintenance release
August 2014	2014.08.18	Initial release

Instantiating the HPS Component 27

2016.10.28

a10_5v4 Subscribe Send Feedback

This chapter describes the parameters available in the hard processor system (HPS) component parameter editor, which opens when you add or edit an HPS component. You instantiate the HPS component in Qsys.

Related Information

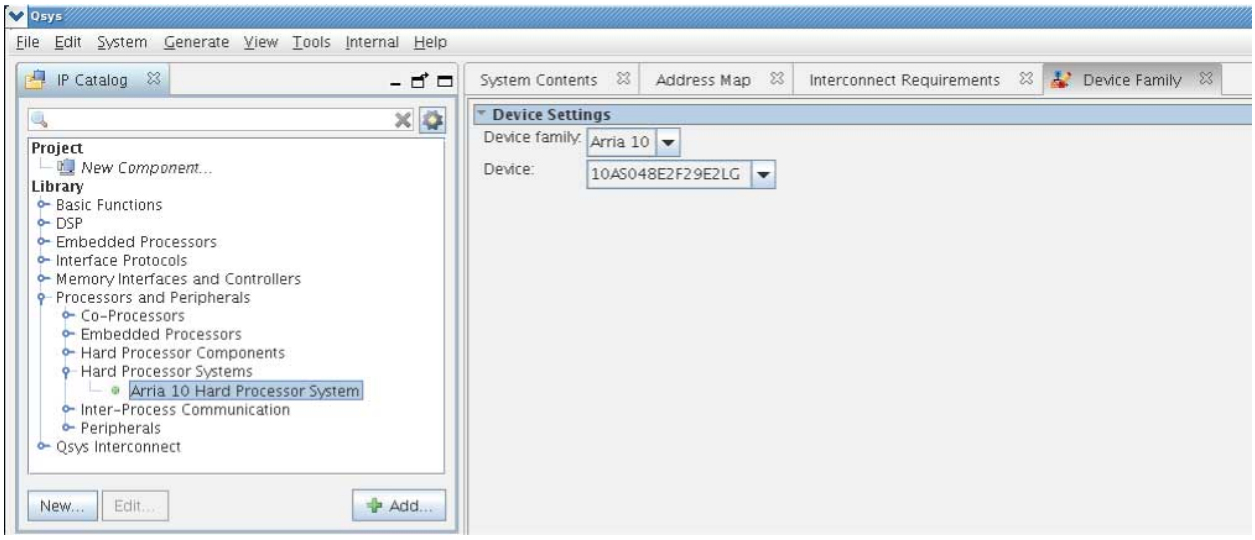
Arria 10 Device Datasheet

The HPS requires specific device targets. For a detailed list of supported devices, refer to the *Arria 10 Device Datasheet* .

Using the HPS Parameter Editor

1. Install the Arria 10 ACDS version 14.0.
2. Open the Quartus Prime software.
3. Open Qsys by selecting **Tools** > **Qsys**.
4. In the **Device Family** tab, select a 10ASxxx device from the **Device** pull-down menu.
5. In the **IP Catalog** tab, under Library, select **Processors and Peripherals** > **Hard Processor Systems** > **Arria 10 Hard Processor System**.

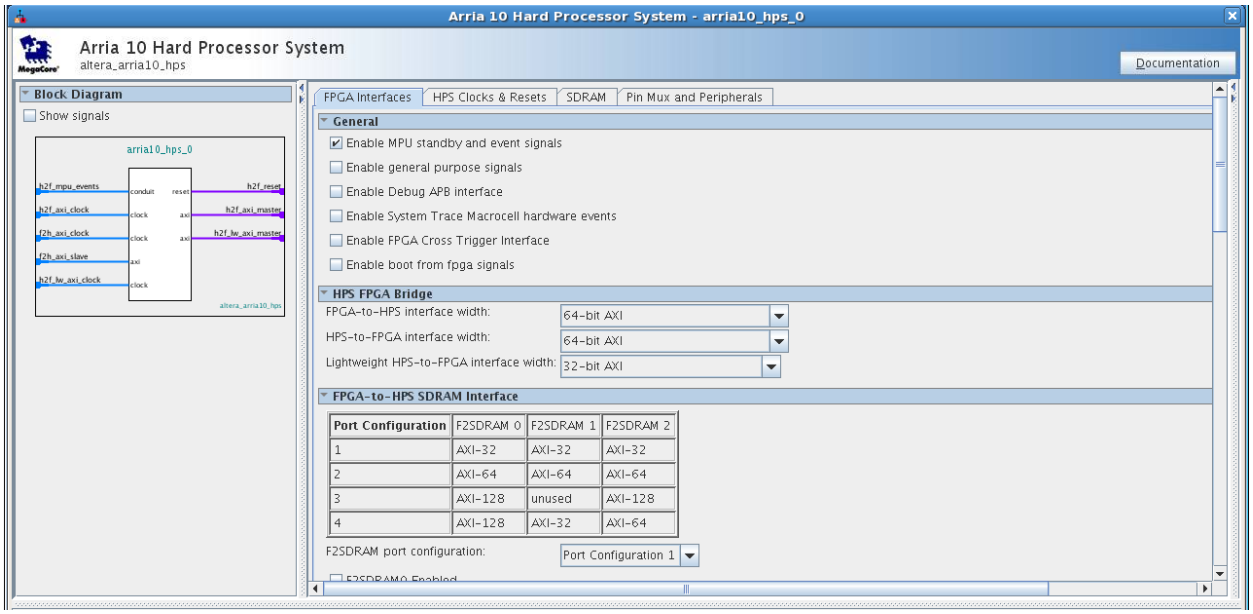
Figure 27-1: Qsys Device Family Tab



FPGA Interfaces

The **FPGA Interfaces** tab is one of four tabs on the HPS Component. This tab contains several groups with the parameters shown in the following figure.

Figure 27-2: FPGA Interfaces Tab



Related Information

Introduction to the HPS Component on page 26-1

General Interfaces

When enabled, the interfaces described in the following table become visible in the HPS component.

Table 27-1: General Parameters

Parameter Name	Parameter Description	Interface Name
Enable MPU standby and event signals	Enables interfaces that perform the following functions: • Notify the FPGA fabric that the microprocessor unit (MPU) is in standby mode. • Wake up an MPCore processor from a wait for event (WFE) state.	`h2f_mpu_events`
	Input for FPGA to signal events to both processors. This signal must be asserted high for at least two MPU clock cycles to be visible to either Cortex-A9 processor core. This signal is used to wake either processor core from standby mode.	`h2f_mpu_eventi`
	Output event from either processor sent to the FPGA. The output event is a multiple cycle pulse so logic in the FPGA should detect it using a rising edge detector circuit to detect the occurrence of the event. This signal is asserted each time either processor executes the SEV instruction.	`h2f_mpu_evento`
	Output per processor that indicates if the processor is in WFE standby mode. When high the processor is in WFE standby mode.	`h2f_mpu_standbywfe [1..0]`
	Output per processor that indicates if the processor is in WFI standby mode. When high the processor is in WFI standby mode.	`h2f_mpu_standbywfi [1..0]`
Enable general purpose signals	Enables a pair of 32-bit unidirectional general-purpose interfaces between the FPGA fabric and the FPGA manager in the HPS portion of the SoC device.	`h2f_gp` • `h2f_gp_in [31..0]` • `h2f_gp_out [31..0]`

Parameter Name	Parameter Description	Interface Name

Enable Debug APB interface	Enables debug interface to the FPGA, allowing access to debug components in the HPS. For more information, refer to the *CoreSight Debug and Trace* chapter.	`h2f_debug_apb`
		• `h2f_dbg_apb_PADDR [17..0]`
		• `h2f_dbg_apb_PADDR31`
		• `h2f_dbg_apb_PENABLE`
		• `h2f_dbg_apb_PRDATA [31..0]`
		• `h2f_dbg_apb_PREADY`
		• `h2f_dbg_apb_PSEL`
		• `h2f_dbg_apb_PSLVERB`
		• `h2f_dbg_apb_PWDATA [31..0]`
		• `h2f_dbg_apb_PWRITE`
		`h2f_debug_apb_sideband`
		• `h2f_debug_apb_PCLKEN`
		• `h2f_debug_apb_DBG_APB_DISABLE`
		`h2f_debug_apb_clock`
		• `h2f_abg_apb_clk`
		`h2f_debug_apb_reset`
		• `h2f_dbg_apb_rst_n`

Enable System Trace Macrocell hardware events	Enables system trace macrocell (STM) hardware events, allowing logic inside the FPGA to insert messages into the trace stream. For more information, refer to the *CoreSight Debug and Trace* chapter.	`f2h_stm_hw_events` • `f2h_stm_hwevents [27..0]`
Enable FPGA Cross Trigger interface	Enables the cross trigger interface (CTI), which allows trigger sources and sinks to interface with the embedded cross trigger (ECT). For more information, refer to the *CoreSight Debug and Trace* chapter. If this interface has to be connected to a SignalTap II instance in the FPGA fabric, then it has to be left disabled in Qsys.	`h2f_cti` • `h2f_cti_trig_in [7..0]` • `h2f_cti_trig_out_ack [7..0]` • `h2f_cti_trig_out [7..0]` • `h2f_cti_trig_in_ack [7..0]`

Parameter Name	Parameter Description	Interface Name
Enable boot from FPGA signals	Enables an input to the HPS indicating whether a preloader is available in the on-chip memory of the FPGA. This option also enables a separate input to the HPS indicating a fallback preloader is available in the FPGA memory. A fallback preloader is used when there is no valid preloader image found in flash memory. For more information, refer to *Appendix A: Booting and Configuration*.	`f2h_boot_from_fpga`
	This is an active high signal which the Boot ROM polls to determine when the FPGA is configured and the memory located at offset 0x0 from the FPGA to HPS bridge is ready to be written to. When the FPGA is not configured the hardware will drive this signal low. You will need to drive this signal high when the memory in the FPGA is ready to accept memory mapped transactions. The `f2h_boot_from_fpga_ready` signal is used by the Boot ROM when accessing the public key stored in the FPGA. The Boot ROM will only use the signal if asserted to boot with the public key. For more information, refer to *Appendix A: Booting and Configuration*.	`f2h_boot_from_fpga_ready`
	This is an active high signal which the Boot ROM polls when all the preloaders fail to load. This signal is driven low when the FPGA is not configured. You will need to drive the signal high in your design. The Boot ROM will continuously poll both **f2h_boot_from_fpga** signals until both are set. For more information, refer to *Appendix A: Booting and Configuration*.	`f2h_boot_from_fpga_on_failure`

Related Information

- **CoreSight Debug and Trace** on page 10-1
  Enabling the TPIU exposes trace signals to the device pins. Refer to the *CoreSight Debug and Trace* for more information.
- **Booting and Configuration** on page 30-1
  For detailed information about the HPS boot sequence, refer to the *Booting and Configuration*.

FPGA-to-HPS SDRAM Interface

In the FPGA-to-HPS SDRAM interface table, use the F2SDRAM port configuration pull-down menu to choose bus widths for each SDRAM. You can add one to three SDRAM ports that make the HPS SDRAM subsystem accessible to the FPGA fabric.

The following table shows the width options for the SDRAM ports in Qsys.

Table 27-2: FPGA-to-HPS SDRAM Port and Interface Names

Port configuration	F2SDRAM 0	F2SDRAM 1	F2SDRAM 2
1	AXI-32	AXI-32	AXI-32
2	AXI-64	AXI-64	AXI-64
3	AXI-128	unused	AXI-128
4	AXI-128	AXI-32	AXI-64

Table 27-3: FPGA-to-HPS SDRAM Port and Interface Names

Port Name	Interface Name
F2SDRAM 0	`f2sdram0_data` `f2sdram0_clock`
F2SDRAM 1	`f2sdram1_data` `f2sdram1_clock`
F2SDRAM 2	`f2sdram2_data` `f2sdram2_clock`

Note: You can configure the slave interface to a data width of 32-, 64-, 128-bits. To facilitate accessing this slave from a memory-mapped master with a smaller address width, you can use the *Altera Address Span Extender*.

Related Information

Using the Address Span Extender Component on page 27-17
Address span extender details

DMA Peripheral Request

You can enable each direct memory access (DMA) controller peripheral request ID individually. Each request ID enables an interface for FPGA soft logic to request one of eight logical DMA channels to the FPGA.

Related Information

DMA Controller on page 16-1
For more information, refer to the *DMA Controller* chapter in the *Arria 10 Hard Processor System Technical Reference Manual* .

Security Manager

Table 27-4: Anti Tamper signals

Interface Name	Description	Associated Signal
`h2f_security`	The security manager anti-tamper interface provides communication between the anti-tamper logic in the FPGA and the HPS security manager.	`f2h_security_anti_tamper_in` `f2h_security_anti_tamper_out`

Related Information

SoC Security on page 6-1

Interrupts

Table 27-5: FPGA-to-HPS Interrupts Interface

Parameter Name	Parameter Description	Interface Name
Enable FPGA-to-HPS Interrupts	Enables interface for FPGA interrupt signals to the MPU (in the HPS).	f2h_irq0 f2h_irq1

You can enable the interfaces for each HPS peripheral interrupt to the FPGA.

Table 27-6: HPS-to-FPGA Interrupts Interface

The following table lists the available HPS to FPGA interrupt interfaces and the corresponding parameters to enable them.

Parameter Name	Parameter Description	Interface Name
Enable Clock Peripheral Interrupts	Enables interface for HPS clock manager and MPU wake-up interrupt signals to the FPGA	`h2f_clkmgr_interrupt` `h2f_mpuwakeup_interrupt`

Parameter Name	Parameter Description	Interface Name
Enable CTI Interrupts	Enables interface for HPS cross-trigger interrupt signals to the FPGA	h2f_ncti_interrupt0 h2f_ncti_interrupt1
Enable DMA Interrupts	Enables interface for HPS DMA channels interrupt and DMA abort interrupt to the FPGA	h2f_dma_interrupt0 h2f_dma_interrupt1 h2f_dma_interrupt2 h2f_dma_interrupt3 h2f_dma_interrupt4 h2f_dma_interrupt5 h2f_dma_interrupt6 h2f_dma_interrupt7 h2f_dma_abort_interrupt
Enable EMAC Interrupts	Enables interface for HPS Ethernet MAC controller interrupt to the FPGA. EMAC must be enabled in Pin Mux Tab before enabling interrupt.	h2f_emac0_interrupt h2f_emac1_interrupt h2f_emac2_interrupt
Enable FPGA manager Interrupts	Enables interface for the HPS FPGA manager interrupt to the FPGA	h2f_fpga_man_interrupt
Enable GPIO Interrupts	Enables interface for the HPS general purpose IO (GPIO) interrupt to the FPGA	h2f_gpio0_interrupt h2f_gpio1_interrupt h2f_gpio2_interrupt
Enable HMC Interrupts	Enable the HPS peripheral interrupt for HMC to be driven into the FPGA fabric.	h2f_hmc_interrupt
Enable I2C-EMAC Interrupts	Enable the HPS peripheral interrupt for I2CEMAC to be driven into the FPGA fabric	h2f_i2c_emac0_interrupt h2f_i2c_emac1_interrupt h2f_i2c_emac2_interrupt
Enable I2C Peripherals Interrupts	Enable the HPS peripheral interrupt for I2C0 to be driven into the FPGA fabric. The I2C must be enabled in the Pin Mux Tab before enabling interrupt.	h2f_i2c0_interrupt h2f_i2c1_interrupt

Parameter Name	Parameter Description	Interface Name
Enable L4 Timer Interrupts	Enables the HPS peripheral interrupt for L4TIMER to be driven into the FPGA fabric.	`h2f_timer_l4sp_0_interrupt` `h2f_timer_l4sp_1_interrupt`
Enable NAND Interrupts	Enables interface for the HPS NAND controller interrupt to the FPGA. The NAND IP Block must be enabled in Pin Mux Tab before enabling interrupt.	`h2f_nand_interrupt`
Enable Quad SPI Interrupts	Enables interface for the HPS QSPI controller interrupt to the FPGA. The Quad SPI IP Block must be enabled in Pin Mux Tab before enabling interrupt.	`h2f_qspi_interrupt`
Enable SYS Timer Interrupts	Enables the HPS peripheral interrupt for SYSTIMER to be driven into the FPGA fabric.	`h2f_timer_sys_0_interrupt` `h2f_timer_sys_1_interrupt`
Enable SD/MMC Interrupts	Enables interface for the HPS SD/MMC controller interrupt to the FPGA. The SD/MMC IP Block must be enabled in Pin Mux Tab before enabling interrupt.	`h2f_sdmmc_interrupt`
Enable SPI Master Interrupts	Enables interface for the HPS SPI master controller interrupt to the FPGA. The SPI Master IP Block must be enabled in Pin Mux Tab before enabling interrupt.	`h2f_spim0_interrupt` `h2f_spim1_interrupt`
Enable SPI Slave Interrupts	Enables interface for the HPS SPI slave controller interrupt to the FPGA. The SPI IP Block must be enabled in Pin Mux Tab before enabling interrupt.	`h2f_spis0_interrupt` `h2f_spis1_interrupt`

Enable ECC/Parity_L1 Interrupts	Enables the HPS peripheral interrupt for ECC single and double bit error and L1 parity error to be driven into the FPGA fabric.	`h2f_ecc_serr_interrupt`
		`h2f_ecc_derr_interrupt`
		`h2f_parity_l1_interrupt`

Parameter Name	Parameter Description	Interface Name
Enable UART Interrupts	Enables interface for the HPS UART controller interrupt to the FPGA. The UART IP Block must be enabled in Pin Mux Tab before enabling interrupt.	`h2f_uart0_interrupt` `h2f_uart1_interrupt`
Enable USB Interrupts	Enables interface for the HPS USB controller interrupt to the FPGA. The USB IP Block must be enabled in Pin Mux Tab before enabling interrupt.	`h2f_usb0_interrupt` `s2f_usb1_interrupt`
Enable Watchdog Interrupts	Enables interface for the HPS watchdog interrupt to the FPGA	`h2f_wdog0_interrupt` `h2f_wdog1_interrupt`

AXI Bridges

Table 27-7: Bridge Parameters

Parameter Name	Parameter Description	Interface Name
FPGA-to-HPS interface width	Enable or disable the FPGA-to-HPS interface; if enabled, set the data width to 32, 64, or 128 bits.	`f2h_axi_slave` `f2h_axi_clock`
HPS-to-FPGA interface width	Enable or disable the HPS-to-FPGA interface; if enabled, set the data width to 32, 64, or 128 bits.	`h2f_axi_master` `h2f_axi_clock`
Lightweight HPS-to-FPGA interface width	Enable or disable the lightweight HPS-to-FPGA interface. When enabled, the data width is 32 bits.	`h2f_lw_axi_master` `h2f_lw_axi_clock`

Note: To facilitate accessing these slaves from a memory-mapped master with a smaller address width, you can use the Altera Address Span Extender.

Related Information

Using the Address Span Extender Component on page 27-17
Address span extender details

Configuring HPS Clocks and Resets

The **HPS Clocks and Reset** tab is one of four tabs on the HPS Component. This tab contains several groups with the following parameters:

Alternate Clock Source from FPGA on page 27-11

User Clocks on page 27-11

Reset Interfaces on page 27-12

Peripheral FPGA Clocks on page 27-13

Alternate Clock Source from FPGA

Table 27-8: Alternate Clock Source Parameters

Parameter Name	Parameter Description	Clock Interface Name
Enable Alternate Clock Source from FPGA	Drives the HPS Clock source to originate from the FPGA fabric instead of the dedicated HPS input clock pin.	`f2h_free_clk`

User Clocks

When you enable a HPS-to-FPGA user clock, you must manually enter its maximum frequency for timing analysis. The TimeQuest Timing Analyzer has no other information about how software running on the HPS configures the phase-locked loop (PLL) outputs. Each possible clock, including clocks that are available from peripherals, has its own parameter for describing the clock frequency.

User Clock Parameters

The frequencies that you provide are the maximum expected frequencies. The actual clock frequencies can be modified through the register interface, for example by software running on the microprocessor unit (MPU). For further details, refer to *Selecting PLL Output Frequency and Phase*.

Parameter Name	Parameter Description	Clock Interface Name
Enable HPS-to-FPGA user 0 clock	Enable main PLL from HPS-to-FPGA	`h2f_user0_clock`
User 0 clock frequency	Specify the maximum expected frequency for the main PLL	

Parameter Name	Parameter Description	Clock Interface Name
Enable HPS-to-FPGA user 1 clock	Enable peripheral PLL from HPS-to-FPGA	h2f_user1_clock
User 1 clock frequency	Specify the maximum expected frequency for the peripheral PLL	

Clock Frequency Usage

The clock frequencies you provide are reported in a Synopsys Design Constraints File (**.sdc**) generated by Qsys. The **.sdc** file will be referenced in the system **.qip** file when the system is generated.

Related Information

Clock Manager on page 2-1
For general information about clock signals, refer to the *Clock Manager* chapter in the *Arria 10 Hard Processor System Technical Reference Manual* .

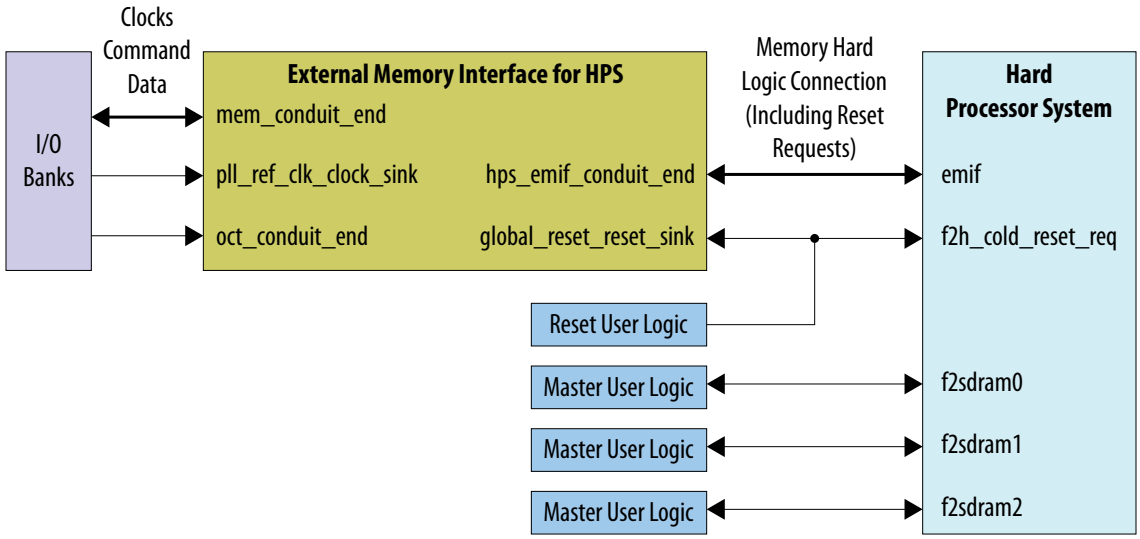
Reset Interfaces

You can enable the resets on an individual basis through the **HPS Clocks and resets** tab under the **Resets** sub tab.

Table 27-9: Reset Parameters

Parameter Name	Parameter Description	Interface Name
Enable HPS-to-FPGA cold reset output	Enable interface for HPS-to-FPGA cold reset output	h2f_cold_reset
Enable HPS warm reset handshake signals	Enable an additional pair of reset handshake signals allowing soft logic to notify the HPS when it is safe to initiate a warm reset in the FPGA fabric.	h2f_warm_reset_handshake
Enable FPGA-to-HPS debug reset request	Enable interface for FPGA-to-HPS debug reset request	f2h_debug_reset_req
Enable FPGA-to-HPS warm reset request	Enable interface for FPGA-to-HPS warm reset request	f2h_warm_reset_req
Enable FPGA-to-HPS cold reset request	Enable interface for FPGA-to-HPS cold reset request	f2h_cold_reset_req

Figure 27-3: Recommended SDRAM Reset Connections



The SDRAM reset relies on the `f2h_cold_reset_req`to be enabled. The reset can be enabled through the Qsys **Resets** tab.

Note: SDRAM and the EMIF conduit can be enabled through the Qsys **FPGA Interface** and **SDRAM** tab.

Related Information

Reset Manager on page 3-1
For more information about the reset interfaces, refer to *Functional Description of the Reset Manager* in the *Reset Manager* chapter in the *Arria 10 Hard Processor System Technical Reference Manual* .

Peripheral FPGA Clocks

Parameter Name	Parameter Description
EMAC 0 (emac0_md_clk clock frequency)	If EMAC 0 peripheral is routed to FPGA, use the input field to specify EMAC 0 MDIO clock frequency
EMAC 0 (emac0_gtx_clk clock frequency)	If EMAC 0 peripheral is routed to FPGA, use the input field to specify EMAC 0 transmit clock frequency
EMAC 1 (emac1_md_clk clock frequency)	If EMAC 1 peripheral is routed to FPGA, use the input field to specify EMAC 1 MDIO clock frequency
EMAC 1 (emac1_gtx_clk clock frequency)	If EMAC 1 peripheral is routed to FPGA, use the input field to specify EMAC 1 transmit clock frequency
EMAC 2 (emac2_md_clk clock frequency)	If EMAC 2 peripheral is routed to FPGA, use the input field to specify EMAC 2 MDIO clock frequency

Parameter Name	Parameter Description
EMAC 2 (emac2_gtx_clk clock frequency)	If EMAC 2 peripheral is routed to FPGA, use the input field to specify EMAC 2 transmit clock frequency
QSPI (qspi_sclk_out clock frequency)	If QSPI peripheral is routed to FPGA, use the input field to specify QSPI serial clock frequency
SDMMC (sdmmc_cclk)	If this peripheral pin multiplexing is configured to route to FPGA fabric, use the input field to specify the SDMMC sdmmc_cclk clock frequency
SPIM 0 (spim0_sclk_out clock frequency)	If SPI master 0 peripheral is routed to FPGA, use the input field to specify SPI master 0 output clock frequency
SPIM 1 (spim1_sclk_out clock frequency)	If SPI master 1 peripheral is routed to FPGA, use the input field to specify SPI master 1 output clock frequency
I^2C0 (i2c0_clk clock frequency)	If I^2C 0 peripheral is routed to FPGA, use the input field to specify I^2C 0 output clock frequency
I^2C1 (i2c1_clk clock frequency)	If I^2C 1 peripheral is routed to FPGA, use the input field to specify I^2C 1 output clock frequency
I2CEMAC0 (i2cemac0_clk)	If this peripheral pin multiplexing is configured to route to the FPGA fabric, use the input field to specify the I2CEMAC0 i2cemac0_clk clock frequency
I2CEMAC1 (i2cemac1_clk)	If this peripheral pin multiplexing is configured to route to the FPGA fabric, use the input field to specify the I2CEMAC1 i2cemac1_clk clock frequency
I2CEMAC2 (i2cemac2_clk)	If this peripheral pin multiplexing is configured to route to the FPGA fabric, use the input field to specify the I2CEMAC2 i2cemac2_clk clock frequency

Configuring Peripheral Pin Multiplexing

The **Pin Mux and Peripherals** tab is one of four tabs on the HPS Component. This tab contains several groups with the following parameters:

Configuring Peripherals on page 27-14

Connecting Unassigned Pins to GPIO on page 27-16

Peripheral Signals Routed to FPGA on page 27-16

Configuring Peripherals

The **Pin MUX and Peripherals** tab contains three sub tabs **IP Selection**, **Advanced Pin Placement**, and **Advanced FPGA Placement**. The **IP Selection** tab contains a list of peripherals that can be enabled and

either routed to the HPS I/Os or to the FPGA. You can enable one or more instances of each peripheral type by using the drop down menu next to each peripheral. When enabled, some peripherals also have mode settings specific to their functions.

Note: Once you have selected a peripheral, you must click the **Apply Selections** button in order to enable the selected peripherals.

Under the the **IP Selection** tab, Qsys allows you to boot from the following flash device sources if selected.

- NAND
- SD/MMC
- QSPI

In the **Advanced Pin Placement** tab you can be more specific about the placement of each peripheral in the HPS dedicated I/O and HPS shared I/O quadrant space. Each location has a pull down IP selection menu where you can select a peripheral for the location.

Pin placement for peripherals cannot span both dedicated I/O and HPS shared I/O, except for the following situations:

- QSPI signals `qspi_ss_o[2]` and `qspi_ss_o[3]` are available to the HPS shared I/O and can be routed to any quadrant selected.
- If the SD/MMC signals are routed to the HPS shared I/O, and the `SDMMC_PWR_ENA` pin is required, then it is only available to the HPS dedicated I/O.
- If the signals for all three EMACs are routed to the HPS shared I/O, the MDIO signals for all three EMACs can be routed to the HPS dedicated I/Os.
- If the NAND Flash Controller pins are routed to the HPS dedicated I/O, `NAND_WP_N` is only available to the HPS shared I/O.

In addition, peripheral pins cannot span multiple quadrants within the HPS shared I/O. All pins of an instance must be self-contained in one quadrant, except for NANDx16 and the GPIO modules because of the number of pins required for these peripherals. For example, NANDx16 may be routed to either quadrant 1 and quadrant 2 or quadrant 3 and quadrant 4.

The **Advanced FPGA Placement** tab allows you to route peripherals to the FPGA. Aside from the **IP Selection** tab this tab allows you to focus on the FPGA routing only. There are also SDMMC and NAND Bit-width options. You can select the EMAC interface and PHY options on this tab if peripheral is selected.

Note: Changes in the **IP Selection** tab carry over to the **Advanced Pin Placement** tab, **Advanced FPGA Placement** tab, and vise versa.

You can enable the following types of peripherals:

Related Information

- **CoreSight Debug and Trace** on page 10-1
  Enabling the TPIU exposes trace signals to the device pins. Refer to the *CoreSight Debug and Trace* for more information.
- **NAND Flash Controller** on page 13-1
- **SD/MMC Controller** on page 14-1
  Secure Digital / MultiMediaCard (SD/MMC) Controller chapter in the *Arria 10 Hard Processor System Technical Reference Manual* .
- **Quad SPI Flash Controller** on page 15-1
  Quad serial peripheral interface (SPI) Flash Controller chapter in the *Arria 10 Hard Processor System Technical Reference Manual* .

- **Ethernet Media Access Controller** on page 17-1
  Ethernet Media Access Controller chapter in the *Arria 10 Hard Processor System Technical Reference Manual* .
- **USB 2.0 OTG Controller** on page 18-1
- **SPI Controller** on page 19-1
- **I2C Controller** on page 20-1
- **UART Controller** on page 21-1

Boot Source

Under the **IP Selection** tab you have the option to choose a nonvolatile flash-based memory to be the boot source. The following flash controller peripherals under the IP block list in the **IP Selection** tab have a **Boot** enable button beside them where you can make this selection.

- NAND
- SD/MMC
- QSPI

Related Information

Booting and Configuration on page 30-1
For detailed information about the HPS boot sequence, refer to the *Booting and Configuration*.

Connecting Unassigned Pins to GPIO

For pins that are not assigned to any peripherals, you can assign them to the GPIO peripheral by clicking on the corresponding GPIO push button in the table. Click on the selected push button to remove GPIO pin assignment. The table also shows the GPIO bit number that correlates to the I/O pins.

Peripheral Signals Routed to FPGA

You can route the peripheral signals to the FPGA fabric and assign them to the FPGA I/O pins.

The following steps show you how to enable the peripheral signals:

1. Select a peripheral in the IP Selection list.
2. Choose the To FPGA drop-down box next to the IP you have selected and choose the amount of instances for the peripheral
3. Click the Apply Selections button at the bottom of the peripherals list.

Configuring the External Memory Interface

The **SDRAM** tab is one of four tabs on the HPS component. Enable the conduit to connect to the Arria 10 External Memory Interface. This option enables a conduit interface for the HPS to talk to the Arria 10 External Memory interfaces for the HPS.

The HPS supports one memory interface implementing double data rate 3 (DDR3), double data rate 3 Low Voltage (DDR3L), and double data rate 4 (DDR4) protocols.

Related Information

External Memory Interface In Arria 10 Devices
For more information on the A10 External Memory Interface, refer to the External Memory Interfaces in Arria 10 Devices in the Arria 10 Core Fabric and General Purpose I/O Handbook.

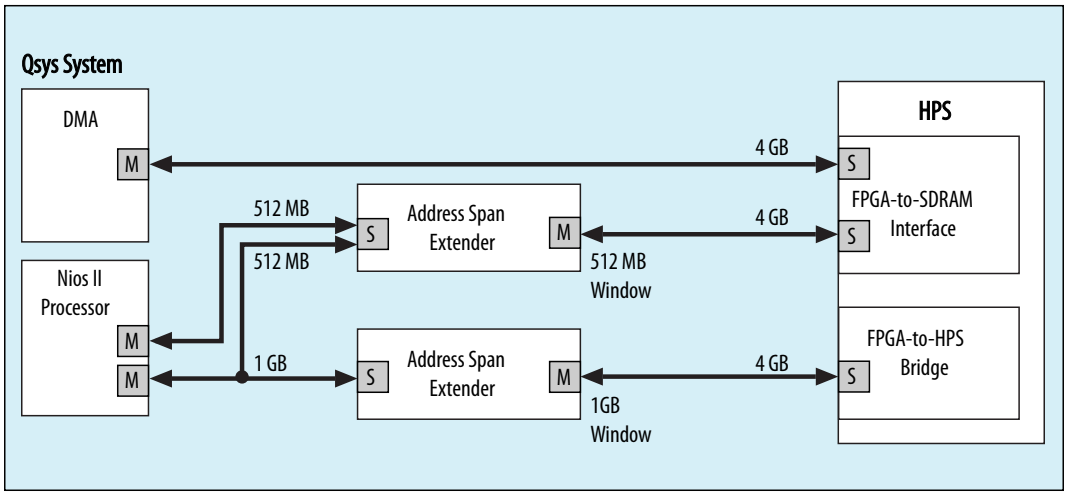
Using the Address Span Extender Component

The FPGA-to-HPS bridge and FPGA-to-HPS SDRAM memory-mapped interfaces expose their entire 4-GB address spaces to the FPGA fabric. The address span extender component provides a memory-mapped window into the address space that it masters. Using the address span extender, you can expose portions of the HPS memory space without needing to expose the entire 4 GB address space.

You can use the address span extender between a soft logic master and an FPGA-to-HPS bridge or FPGA-to-HPS SDRAM interface. This component reduces the number of address bits required for a master to address a memory-mapped slave interface located in the HPS.

Figure 27-4: Address Span Extender Components

Two address span extender components used in a system with the HPS.



You can also use the address span extender in the HPS-to-FPGA direction, for slave interfaces in the FPGA. In this case, the HPS-to-FPGA bridge exposes a limited, variable address space in the FPGA, which can be paged in using the address span extender.

For example, suppose that the HPS-to-FPGA bridge has a 1-GB span, and the HPS needs to access three independent 1-GB memories in the FPGA portion of the device. To achieve this, the HPS programs the address span extender to access one SDRAM (1-GB) in the FPGA at a time. This technique is commonly called paging or windowing.

Related Information

Qsys System Design Components

For more information about the Altera Span Extender, refer to the Address Span Exdenter section in the Qsys System Design Components chapter of the Qsys Design Handbook.

Generating and Compiling the HPS Component

The process of generating and compiling an HPS design is very similar to the process for any other Qsys project. Perform the following steps:

1. Generate the design with Qsys. The generated files include an **.sdc** file containing clock timing constraints. If simulation is enabled, simulation files are also generated.
2. Add *<qsys_system_name>.qip* to the Quartus Prime project. *<qsys_system_name>.qip* is the Quartus Prime IP File for the HPS component, generated by Qsys.

   Note: Qsys generates pin assignments in the **qip** file.
3. Perform analysis and synthesis with the Quartus Prime software.
4. Compile the design with the Quartus Prime software.
5. Optionally back-annotate the SDRAM pin assignments, to eliminate pin assignment warnings the next time you compile the design.

Related Information

- **Configuring Peripheral Pin Multiplexing** on page 27-14
- **Setting Up the HPS Component for Simulation** on page 29-3
  For a description of the simulation files generated, refer to "Simulation Flow" in the Simulating the HPS Component chapter of the *Arria 10 Hard Processor System Technical Reference Manual* .

Document Revision History

Table 27-10: Document Revision History

Date	Version	Changes
October 2016	2016.10.28	Maintenance release.
May 2016	2016.05.27	Removed *FPGA EMAC Switch Interface* section. The application interface (also called the switch interface) is not supported.
May 2016	2016.05.03	Maintenance release.
November 2015	2015.11.02	Added content regarding peripheral pin placmement in HPS shared and dedicated I/O to the "Configuring Peripherals" section.
May 2015	2015.05.04	• Updated "Reset Interfaces" section. • Updated "General Interfaces" section.
December 2014	2014.12.15	Initial release.

HPS Component Interfaces 28

2016.10.28

a10_5v4

This chapter describes the interfaces, including clocks and resets, implemented by the hard processor system (HPS) component.

The majority of the resets can be enabled on an individual basis. The exception is the `h2f_reset` interface, which is always enabled.

You must declare the clock frequency of each HPS-to-FPGA clock for timing purposes. Each possible clock, including ones that are available from peripherals, has its own parameter for describing the clock frequency. Declaring the clock frequency for HPS-to-FPGA clocks specifies how you plan to configure the PLLs and peripherals, to enable TimeQuest to accurately estimate system timing. It has no effect on PLL settings.

Related Information

- **HPS Component Interfaces** on page 28-1
  For information about instantiating the HPS component, refer to the *Instantiating the HPS Component* chapter.
- **Info Center**
  For Advanced Microcontroller Bus Architecture (AMBA) Advanced eXtensible Interface (AXI) protocol timing, refer to the AMBA AXI Protocol Specification v1.0, which you can download from the ARM info center website.

Memory-Mapped Interfaces

FPGA-to-HPS Bridge

Interface Name	Description	Associated Clock Interface
`f2h_axi_slave`	FPGA-to-HPS AXI slave interface	`f2h_axi_clock`

The FPGA-to-HPS interface is a configurable data width AXI slave allowing FPGA masters to issue transactions to the HPS. This interface allows the FPGA fabric to access the majority of the HPS slaves. This interface also provides a coherent memory interface.

Other interface standards in the FPGA fabric, such as connecting to Avalon® Memory-Mapped (Avalon-MM) interfaces, can be supported through the use of soft logic adapters. The Qsys system integration tool automatically generates adapter logic to connect AXI to Avalon-MM interfaces.

The FPGA-to-HPS interface is an AXI-3 compliant interface with the following features:

- Configurable data width: 32, 64, or 128 bits
- Ready latency support data widths: 32, 64, or 128 bits

Related Information

- **Clocks** on page 28-4
- **Features of the HPS-FPGA Bridges** on page 8-1
  For more information, refer to the *HPS FPGA AXI Bridges Component* chapter.
- **HPS Component Interfaces** on page 28-1
  For information about the address span extender, refer to "Using the Address Span Extender Component" in the *Instantiating the HPS Component* chapter.

HPS-to-FPGA and Lightweight HPS-to-FPGA Bridges

Table 28-1: HPS-to-FPGA and Lightweight HPS-to-FPGA Bridges and Clocks

Interface Name	Description	Associated Clock Interface
`h2f_axi_master`	HPS-to-FPGA AXI master interface	`h2f_axi_clock`
`h2f_lw_axi_master`	HPS-to-FPGA lightweight AXI master interface	`h2f_lw_axi_clock`

The HPS-to-FPGA interface is a configurable data width AXI master (32-, 64-, or 128-bit) that allows HPS masters to issue transactions to the FPGA fabric.

The lightweight HPS-to-FPGA interface is a 32-bit AXI master that allows HPS masters to issue transactions to the FPGA fabric.

Other interface standards in the FPGA fabric, such as connecting to Avalon-MM interfaces, can be supported through the use of soft logic adaptors. The Qsys system integration tool automatically generates adaptor logic to connect AXI to Avalon-MM interfaces.

Each AXI bridge accepts a clock input from the FPGA fabric and performs clock domain crossing internally. The exposed AXI interface operates on the same clock domain as the clock supplied by the FPGA fabric.

Latency Support for (32-,64, or 128-bit)

Related Information

- **Clocks** on page 28-4
- **Features of the HPS-FPGA Bridges** on page 8-1
  For more information, refer to the *HPS FPGA AXI Bridges Component* chapter.

FPGA-to-HPS SDRAM Interface

The FPGA-to-HPS SDRAM interface is a direct connection between the FPGA fabric and the HPS SDRAM controller. The interface is highly configurable, allowing a mix between the four port configurations and the port width. The FPGA-to-HPS SDRAM interface supports AMBA AXI interface standards.

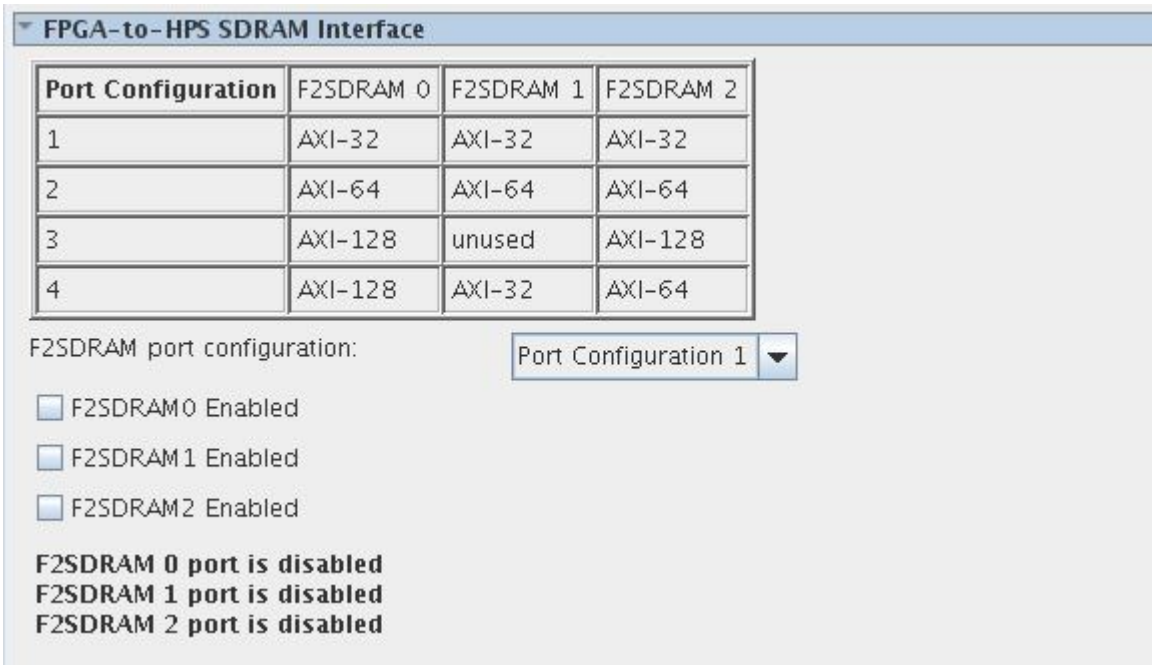
Table 28-2: FPGA-to-HPS SDRAM Interfaces and Clocks

Interface Name	Associated Data Interface	Associated Clock Interface
F2SDRAM0	f2sdram0_data	f2sdram0_clock
F2SDRAM1	f2sdram1_data	f2sdram1_clock
F2SDRAM2	f2sdram2_data	f2sdram2_clcok

The FPGA-to-HPS SDRAM interface is a configurable interface to the multi-port SDRAM controller. There are four port configurations that are a combination of the SDRAM ports 0 - 2 that you are able to select. Once a port configuration is selected, you can choose to enable the ports that you need.

The total data width of all interfaces is limited to a maximum of 256 bits in the read direction and 256 bits in the write direction. The HPS SDRAM controller supports up to 3 masters (command ports), 3x 64-bit read data ports and 3x 64-bit write data ports.

Figure 28-1: FPGA-to-HPS SDRAM Qsys Interface



Related Information

- Clocks on page 28-4

- **HPS Component Interfaces** on page 28-1
  For information about the address span extender, refer to "Using the Address Span Extender Component" in the *Instantiating the HPS Component* chapter.

EMIF Conduit

Under the SDRAM tab in Qsys, the EMIF conduit can be enabled. This enables the HPS dedicated conduit to the Arria 10 External Interface for the HPS. This conduit cannot connect to any other External Memory Interface (EMIF). Only IP generated by the Arria 10 External Memory Interface for HPS Qsys library should be used.

Clocks

The HPS-to-FPGA clock interface supplies physical clocks to the FPGA. These clocks and resets are generated in the HPS.

Alternative Clock Inputs to HPS PLLs

This section lists alternative clock inputs to HPS PLLs.

- `f2h_free_clk`—FPGA-to-HPS free clock. Drives the HPS clock source to originate from the FPGA fabric instead of the dedicated HPS input clock pin.

User Clocks

A user clock is a PLL output that is connected to the FPGA fabric rather than the HPS. You can connect a user clock to logic that you instantiate in the FPGA fabric.

- `h2f_user0_clock`—HPS-to_FPGA user clock, driven from main PLL
- `h2f_user1_clock`—HPS-to-FPGA user clock, driven from peripheral PLL

AXI Bridge FPGA Interface Clocks

The AXI interface has an asynchronous clock crossing in the FPGA-to-HPS bridge. The FPGA-to-HPS and HPS-to-FPGA interfaces are synchronized to clocks generated in the FPGA fabric. These interfaces can be asynchronous to one another.

- `f2h_axi_clock`—AXI slave clock for FPGA-to-HPS bridge, generated in FPGA fabric
- `h2f_axi_clock`—AXI master clock for HPS-to-FPGA bridge, generated in FPGA fabric
- `h2f_lw_axi_clock`—AXI master clock for lightweight HPS-to-FPGA bridge, generated in FPGA fabric

SDRAM Clocks

You can configure the HPS component with up to three FPGA-to-HPS SDRAM clocks.

Each command channel to the SDRAM controller has an individual clock source from the FPGA fabric. The interface clock is always supplied by the FPGA fabric, with clock crossing occurring on the HPS side of the boundary.

The FPGA-to-HPS SDRAM clocks are driven by soft logic in the FPGA fabric.

- f2h_sdram0_clock—SDRAM clock for port 0
- f2h_sdram1_clock—SDRAM clock for port 1
- f2h_sdram2_clock—SDRAM clock for port 2

Peripheral FPGA Clocks

Table 28-3: Peripheral FPGA Clocks

Clock Name	Description
emac_md_clk	Ethernet PHY management interface clock
emac_gtx_clk	Ethernet transmit clock that is used by the PHY in GMII mode
emac_rx_clk_in	Ethernet MAC reference clock from the PHY
emac_tx_clk_in	Ethernet MAC uses this clock input for TX reference
emac_ptp_ref_clock	Ethernet timestamp precision time protocol (PTP) reference clock
qspi_sclk_out	QSPI master clock output
qspi_s2f_clk	QSPI serial clock output (connects to the FPGA clock network)
sdmmc_clk_in	Clock for SD/MMC controller
sdmmc_cclk	Generated output clock for card
spim_sclk_out	SPI master serial clock output
spis_sclk_in	SPI slave serial clock input
i2c_clk	I^2C outgoing clock (part of the SCL bidirectional pin signals)
i2c_scl_in	I^2C incoming clock (part of the SCL bidirectional pin signals)
i2cemac_clk	I^2C EMAC outgoing clock

Note: The Peripheral FPGA Clocks can only be accessed once the corresponding IP Block has been selected and routed to the FPGA through the Pin Mux and Peripherals tab.

Resets

This section describes the reset interfaces to the HPS component.

Related Information

Reset Manager on page 3-1
For details about the HPS reset sequences, refer to the *Functional Description of the Reset Manager* in the *Reset Manager* chapter .

HPS-to-FPGA Reset Interfaces

The following interfaces allow the HPS to reset soft logic in the FPGA fabric:

- `h2f_reset`—HPS-to-FPGA warm reset
- `h2f_cold_reset`—HPS-to-FPGA cold reset
- `h2f_warm_reset_handshake`—Warm reset request and acknowledge interface between HPS and FPGA

HPS External Reset Request

The following interfaces allow soft logic in the FPGA fabric to request for different types of HPS resets:

- `f2h_cold_reset_req`—FPGA-to-HPS cold reset request
- `f2h_warm_reset_req`—FPGA-to-HPS warm reset request
- `f2h_dbg_reset_req`—FPGA-to-HPS debug reset request

Peripheral Reset Interfaces

The following are Ethernet reset interfaces, that can be used when the Ethernet is routed to the FPGA:

- `emac_tx_reset`— Ethernet transmit clock reset output used to reset external PHY TX clock domain logic
- `emac_rx_reset`— Ethernet receive clock reset output used to reset external PHY RX clock domain logic

Debug and Trace Interfaces

FPGA System Trace Macrocell Events Interface

The system trace macrocell (STM) hardware events allow logic in the FPGA to insert messages into the trace stream.

- `f2h_stm_hw_events`

FPGA Cross Trigger Interface

The cross trigger interface (CTI) allows trigger sources and sinks to interface with the embedded cross trigger (ECT).

- `h2f_cti`

Debug APB Interface

The debug Advanced Peripheral Bus (APB™) interface allows debug components in the FPGA fabric to debug components on the CoreSight™ debug APB.

- `h2f_debug_apb`
- `h2f_debug_apb_sideband`
- `h2f_debug_apb_reset`
- `h2f_debug_apb_clock`

Peripheral Signal Interfaces

Qsys Peripheral Port Interface Mapping

For many of the HPS component peripherals, the Qsys ports can be routed to the HPS pins, FPGA pins or both. The following tables show the available mappings.

NAND Flash Controller Interface

Table 28-4: NAND Flash Controller Interface Qsys Port Mappings

Qsys Port Name	Routed to FPGA	Routed to HPS I/O	HPS Pin Name
`nand_adq_i[15:0]`	Yes	Yes	`NAND_ADQ[15:0]`
`nand_adq_oe`	Yes	Yes	
`nand_adq_o[15:0]`	Yes	Yes	
`nand_ale_o`	Yes	Yes	`NAND_ALE`
`nand_ce_o[3:0]`	Yes, 4 chip enables	Yes, 1 chip enable	`NAND_CE_N`
`nand_cle_o`	Yes	Yes	`NAND_CLE`
`nand_re_o`	Yes	Yes	`NAND_RE_N`
`nand_rdy_busy_i[3:0]`	Yes, 4 ready/busy signals	Yes, 1 ready/busy signal	`NAND_RB`
`nand_we_o`	Yes	Yes	`NAND_WE_N`
`nand_wp_o`	Yes	Yes	`NAND_WP_N`

Related Information

NAND Flash Controller on page 13-1
For more information about the NAND Flash Controller features and interface.

SD/MMC Controller Interface

Table 28-5: SD/MMC Controller Interface Qsys Port Mappings

Qsys Port Name	Routed to FPGA	Routed to HPS I/O	HPS Pin Name
sdmmc_vs_o	Yes	No	-
sdmmc_pwr_ena_o	Yes	Yes	SDMMC_PWR_ENA
sdmmc_wp_i	Yes	No	-
sdmmc_cdn_i	Yes	No	-
sdmmc_rstn_o	Yes	No	-
sdmmc_card_intn_i	Yes	No	-
sdmmc_clk_in_i	Yes	No	-
sdmmc_cclk_out	Yes	Yes	SDMMC_CCLK
sdmmc_cmd_i	Yes	Yes	SDMMC_CMD
sdmmc_cmd_o	Yes	Yes	
sdmmc_cmd_oe	Yes	Yes	
sdmmc_data_i[7:0]	Yes	Yes	SDMMC_DATA[7:0]
sdmmc_data_o[7:0]	Yes	Yes	
sdmmc_data_oe	Yes	Yes	

Related Information

Features of the SD/MMC Controller on page 14-1
For more information about the SD/MMC Controller features and interface.

Quad SPI Flash Controller Interface

Table 28-6: Quad SPI Flash Controller Qsys Port Mappings

Qsys Port Name	Routed to FPGA	Routed to HPS I/O	HPS Pin Name
qspi_sclk_out	Yes	Yes	QSPI_CLK
qspi_s2f_clk	Yes	No	-
qspi_io0_i	Yes	Yes	QSPI_IO0
qspi_io0_o	Yes	Yes	
qspi_mo_oe0	Yes	Yes	
qspi_io1_i	Yes	Yes	QSPI_IO1
qspi_io1_o	Yes	Yes	
qspi_mo_oe1	Yes	Yes	

Qsys Port Name	Routed to FPGA	Routed to HPS I/O	HPS Pin Name
qspi_io2_i	Yes	Yes	QSPI_IO2_WPN
qspi_io2_wpn_o	Yes	Yes	
qspi_mo_oe2	Yes	Yes	
qspi_io3_i	Yes	Yes	QSPI_IO3_HOLD
qspi_io3_hold_o	Yes	Yes	
qspi_mo_oe3	Yes	Yes	
qspi_ss_o[3:0]	Yes	Yes	QSPI_SS[3:0]

Related Information

Quad SPI Flash Controller on page 15-1
For more information about the Quad SPI Flash Controller features and interface.

Ethernet Media Access Controller Interface

Table 28-7: Ethernet Media Access Controller Qsys Port Mappings

Qsys Port Name	Routed to FPGA	Routed to HPS I/O	HPS Pin Name
emac0_phy_txd_o[7:0]	Yes, 8 bits	Yes, 4 bits	EMAC0_TXD[3:0]
emac0_phy_mac_speed_o	Yes	No	-
emac0_phy_txen_o	Yes	Yes	EMAC0_TX_CTL
emac0_phy_txer_o	Yes	No	-
emac0_phy_rxdv_i	No	Yes	EMAC0_RX_CTL
emac0_phy_rxer_i	Yes	No	-
emac0_phy_rxd_i[7:0]	Yes, 8 bits	Yes, 4 bits	EMAC0_RXD[3:0]
emac0_phy_col_i	Yes	No	-
emac0_phy_crs_i	Yes	No	-
emac0_gmii_mdc_o	Yes	Yes	EMAC0_MDC
emac0_gmii_mdo_o	Yes	Yes	EMAC0_MDIO
emac0_gmii_mdo_o_e	Yes	Yes	
emac0_gmii_mdi_i	Yes	Yes	
emac0_ptp_pps_o	Yes	No	-
emac0_ptp_aux_ts_trig_i	Yes	No	-
emac0_clk_rx_i	Yes	Yes	EMAC0_RX_CLK
emac0_clk_tx_i	Yes	No	-
emac0_phy_txclk_o	Yes	Yes	EMAC0_TX_CLK
emac0_rst_clk_tx_n_o	Yes	No	-

Qsys Port Name	Routed to FPGA	Routed to HPS I/O	HPS Pin Name
emac0_rst_clk_rx_n_o	Yes	No	-
emac1_phy_txd_o[7:0]	Yes, 8 bits	Yes, 4 bits	EMAC1_TXD[3:0]
emac1_phy_mac_speed_o	Yes	No	-
emac1_phy_txen_o	Yes	Yes	EMAC1_TX_CTL
emac1_phy_txer_o	Yes	No	-
emac1_phy_rxdv_i	Yes	Yes	EMAC1_RX_CTL
emac1_phy_rxer_i	Yes	No	-
emac1_phy_rxd_i[7:0]	Yes, 8 bits	Yes, 4 bits	EMAC1_RXD[3:0]
emac1_phy_col_i	Yes	No	-
emac1_phy_crs_i	Yes	No	-
emac1_gmii_mdc_o	Yes	Yes	EMAC1_MDC
emac1_gmii_mdo_o	Yes	Yes	EMAC1_MDIO
emac1_gmii_mdo_o_e	Yes	Yes	
emac1_gmii_mdi_i	Yes	Yes	
emac1_ptp_pps_o	Yes	No	-
emac1_ptp_aux_ts_trig_i	Yes	No	-
emac1_clk_rx_i	Yes	Yes	EMAC1_RX_CLK
emac1_clk_tx_i	Yes	No	-
emac1_phy_txclk_o	Yes	Yes	EMAC1_TX_CLK
emac1_rst_clk_tx_n_o	Yes	No	-
emac1_rst_clk_rx_n_o	Yes	No	-
emac2_phy_txd_o[7:0]	Yes, 8 bits	Yes, 4 bits	EMAC2_TXD[3:0]
emac2_phy_mac_speed_o	Yes	No	-
emac2_phy_txen_o	Yes	Yes	EMAC2_TX_CTL
emac2_phy_txer_o	Yes	No	-
emac2_phy_rxdv_i	Yes	Yes	EMAC2_RX_CTL
emac2_phy_rxer_i	Yes	-	No
emac2_phy_rxd_i[7:0]	Yes, 8 bits	Yes, 4 bits	EMAC2_RXD[3:0]
emac2_phy_col_i	Yes	No	-
emac2_phy_crs_i	Yes	No	-
emac2_gmii_mdc_o	Yes	Yes	EMAC2_MDC
emac2_gmii_mdo_o	Yes	Yes	EMAC2_MDIO
emac2_gmii_mdo_o_e	Yes	Yes	
emac2_gmii_mdi_i	Yes	Yes	

Qsys Port Name	Routed to FPGA	Routed to HPS I/O	HPS Pin Name
emac2_ptp_pps_o	Yes	No	-
emac2_ptp_aux_ts_trig_i	Yes	No	_
emac2_clk_rx_i	Yes	Yes	EMAC2_RX_CLK
emac2_clk_tx_i	Yes	No	-
emac2_phy_txclk_o	Yes	Yes	EMAC2_TX_CLK
emac2_rst_clk_tx_n_o	Yes	No	-
emac2_rst_clk_rx_n_o	Yes	No	-

Related Information

Ethernet Media Access Controller on page 17-1
For more information about the Ethernet Media Access Controller features and interface.

USB 2.0 OTG Controller Interface

Table 28-8: USB 2.0 OTG Controller Qsys Port Mappings

Qsys Port Name	Routed to FPGA	Routed to HPS I/O	HPS Pin Name
usb0_ulpi_clk	No	Yes	USB0_CLK
usb0_ulpi_dir	No	Yes	USB0_DIR
usb0_ulpi_nxt	No	Yes	USB0_NXT
usb0_ulpi_stp	No	Yes	USB0_STP
usb0_ulpi_data_i[7:0]	No	Yes	USB0_DATA[7:0]
usb0_ulpi_data_o[7:0]	No	Yes	
usb0_ulpi_data_oe[7:0]	No	Yes	
usb1_ulpi_clk	No	Yes	USB1_CLK
usb1_ulpi_dir	No	Yes	USB1_DIR
usb1_ulpi_nxt	No	Yes	USB1_NXT
usb1_ulpi_stp	No	Yes	USB1_STP
usb1_ulpi_data_i[7:0]	No	Yes	USB1_DATA[7:0]
usb1_ulpi_data_o[7:0]	No	Yes	
usb1_ulpi_data_oe[7:0]	No	Yes	

SPI Controller Interface

Table 28-9: SPI Controller Interface Qsys Port Mappings

Qsys Port Name	Routed to FPGA	Routed to HPS I/O	HPS Pin Name
spim0_sclk_out	Yes	Yes	SPIM0_CLK
spim0_mosi_o	Yes	Yes	SPIM0_MOSI
spim0_mosi_oe	Yes	No	
spim0_miso_i	Yes	Yes	SPIM0_MISO
spim0_ss_in_n	Yes	No	-
spim0_mosi_oe	Yes	No	-
spim0_ss0_n_o	Yes	Yes	SPIM0_SS0_N
spim0_ss1_n_o	Yes	Yes	SPIM0_SS1_N
spim0_ss2_n_o	Yes	No	-
spim0_ss3_n_o	Yes	No	-
spim1_sclk_out	Yes	Yes	SPIM1_CLK
spim1_mosi_o	Yes	Yes	SPIM1_MOSI
spim1_mosi_oe	Yes	Yes	
spim1_miso_i	Yes	Yes	SPIM1_MISO
spim1_ss_in_n	Yes	No	-
spim1_ss0_n_o	Yes	Yes	SPIM1_SS0_N
spim1_ss1_n_o	Yes	Yes	SPIM1_SS1_N
spim1_ss2_n_o	Yes	No	-
spim1_ss3_n_o	Yes	No	-
spis0_sclk_in	Yes	Yes	SPIS0_CLK
spis0_mosi_i	Yes	Yes	SPIS0_MOSI
spis0_ss_in_n	Yes	Yes	SPIS0_SS0_N
spis0_miso_o	Yes	Yes	SPIS0_MISO
spis0_miso_oe	Yes	Yes	
spis1_sclk_in	Yes	Yes	SPIS1_CLK
spis1_mosi_i	Yes	Yes	SPIS1_MOSI
spis1_ss_in_n	Yes	Yes	SPIS1_SS0_N
spis1_miso_o	Yes	Yes	SPIS1_MISO
spis1_miso_oe	Yes	Yes	

Related Information

SPI Controller on page 19-1
For more information about the SPI Controller features and interface.

I2C Controller Interface

Table 28-10: I²C Controller Qsys Port Mappings

Qsys Port Name	Routed to FPGA	Routed to HPS I/O	HPS Pin Name
i2c0_scl_i	Yes	Yes	I2C0_SCL
i2c0_scl_oe	Yes	Yes	
i2c0_sda_i	Yes	Yes	I2C0_SDA
i2c0_sda_oe	Yes	Yes	
i2c1_scl_i	Yes	Yes	I2C1_SCL
i2c1_scl_oe	Yes	Yes	
i2c1_sda_i	Yes	Yes	I2C1_SDA
i2c1_sda_oe	Yes	Yes	
i2c_emac0_scl_i	Yes	Yes	I2C_EMAC0_SCL
i2c_emac0_scl_oe	Yes	Yes	
i2c_emac0_sda_i	Yes	Yes	I2C_EMAC0_SDA
i2c_emac0_sda_oe	Yes	Yes	
i2c_emac1_scl_i	Yes	Yes	I2C_EMAC1_SCL
i2c_emac1_scl_oe	Yes	Yes	
i2c_emac1_sda_i	Yes	Yes	I2C_EMAC1_SDA
i2c_emac1_sda_oe	Yes	Yes	
i2c_emac2_scl_i	Yes	Yes	I2C_EMAC2_SCL
i2c_emac2_scl_oe	Yes	Yes	
i2c_emac2_sda_i	Yes	Yes	I2C_EMAC2_SDA
i2c_emac2_sda_oe	Yes	Yes	

Note: When routing the I²C signals to the FPGA I/O, the I²C output signal should be connected to ground. Refer to the *I²C Controller* chapter for more information.

Related Information

I2C Controller on page 20-1
For more information about the I²C Controller features and interface.

UART Interface

Table 28-11: UART Interface Qsys Port Mappings

Qsys Port Name	Routed to FPGA	Routed to HPS I/O	HPS Pin Name
uart0_cts_n	Yes	Yes	UART0_CTS_N
uart0_dsr_n	Yes	No	-
uart0_dcd_n	Yes	No	-
uart0_ri_n	Yes	No	-
uart0_rx	Yes	Yes	UART0_RX
uart0_dtr_n	Yes	No	-
uart0_rts_n	Yes	Yes	UART0_RTS_N
uart0_out1_n	Yes	No	-
uart0_out2_n	Yes	No	-
uart0_tx	Yes	Yes	UART0_TX
uart1_cts_n	Yes	Yes	UART1_CTS_N
uart1_dsr_n	Yes	No	-
uart1_dcd_n	Yes	No	-
uart1_ri_n	Yes	No	-
uart1_rx	Yes	Yes	UART1_RX
uart1_dtr_n	yes	No	-
uart1_rts_n	Yes	Yes	UART1_RTS_N
uart1_out1_n	Yes	No	-
uart1_out2_n	Yes	No	-
uart1_tx	Yes	Yes	UART1_TX

Related Information

UART Controller on page 21-1
For more information about the UART Controller features and interface.

DMA Controller Interface

The DMA controller interface allows soft IP in the FPGA fabric to communicate with the DMA controller in the HPS. You can configure up to eight separate interface channels.

- f2h_dma0—FPGA DMA controller peripheral request interface 0
- f2h_dma1—FPGA DMA controller peripheral request interface 1
- f2h_dma2—FPGA DMA controller peripheral request interface 2
- f2h_dma3—FPGA DMA controller peripheral request interface 3
- f2h_dma4—FPGA DMA controller peripheral request interface 4

- f2h_dma5—FPGA DMA controller peripheral request interface 5
- f2h_dma6—FPGA DMA controller peripheral request interface 6
- f2h_dma7—FPGA DMA controller peripheral request interface 7

Each of the DMA peripheral request interface contains the following three signals:

- f2h_dma_req—This signal is used to request burst transfer using the DMA
- f2h_dma_single—This signal is used to request single word transfer using the DMA
- f2h_dma_ack—This signal indicates the DMA acknowledgment upon requests from the FPGA

Related Information

DMA Controller on page 16-1
For details about the DMA Controller, refer to *DMA controller* chapter.

Other Interfaces

Related Information

Cortex-A9 MPCore on page 9-4
For more information, refer to *Cortex-A9 MPU Subsystem* .

MPU Standby and Event Interfaces

MPU standby signals are notification signals to the FPGA fabric that the MPU is in standby. Event signals are used to wake up the Cortex-A9 processors from a wait for event (WFE) state. The following shows the signals in the interface:

- h2f_mpu_eventi—Sends an event from logic in the FPGA fabric to the MPU. This FPGA-to-HPS signal is used to wake up a processor that is in a Wait For Event state. Asserting this signal has the same effect as executing the SEV instruction in the Cortex-A9. This signal must be de-asserted until the FPGA fabric is powered-up and configured.
- h2f_mpu_evento—Sends an event from the MPU to logic in the FPGA fabric. This HPS-to-FPGA signal is asserted when an SEV instruction is executed by one of the Cortex-A9 processors.
- h2f_mpu_standbywfe[1:0]—Indicates which Cortex-A9 processor is in the WFE state
- h2f_mpu_standbywfi[1:0]—Indicates which Cortex-A9 processor is in the wait for interrupt (WFI) state

The MPU provides signals to indicate when it is in a standby state. These signals are available to custom hardware designs in the FPGA fabric.

Related Information

Cortex-A9 MPCore on page 9-4
For more information, refer to *Cortex-A9 MPU Subsystem* .

General Purpose Signals

Table 28-12: General Purpose Interfaces

Signal	Description	Associated Signal
h2f_gp	General purpose interface. Enables a pair of 32-bit unidirectional general-purpose interfaces between the FPGA manager in HPS and the FPGA fabric.	h2f_gp_in h2f_gp_out

FPGA-to-HPS Interrupts

You can configure the HPS component to provide 64 general purpose FPGA-to-HPS interrupts, allowing soft IP in the FPGA fabric to trigger interrupts to the MPU's generic interrupt controller (GIC). The interrupts are implemented through the following 32-bit interfaces:

- f2h_irq0—FPGA-to-HPS interrupts 0 through 31
- f2h_irq1—FPGA-to-HPS interrupts 32 through 63

The FPGA-to-HPS interrupts are asynchronous on the FPGA interface. Inside the HPS, the interrupts are synchronized to the MPU's internal peripheral clock (periphclk).

Related Information

HPS Component Interfaces on page 28-1
There are various HPS peripherals to FPGA interrupt interfaces that you can configure. To learn the complete list of interfaces, refer to the *Instantiating the HPS Component* chapter.

Boot from FPGA Interface

You can enable the boot from FPGA interface to indicate the availability of preloader software in the FPGA memory. This interface is used to indicate the availability of a fallback preloader software in FPGA memory as well. The fallback preloader is used when there is no valid preloader image found in HPS flash memories.

Security Manager

Table 28-13: Anti Tamper signal

Interface Name	Description
h2f_security	HPS to FPGA input to trigger memory scrambling of all internal memories.

Document Revision History

Table 28-14: Document Revision History

Date	Version	Changes
October 2016	2016.10.28	Maintenance release.
May 2016	2016.05.27	Removed *FPGA EMAC Switch Interface* section. The application interface (also called the switch interface) is not supported.
May 2016	2016.05.03	Maintenance release.
November 2015	2015.11.02	• Added "Qsys Port Interface Mapping" section and subsections to "Peripheral Signal Interfaces"
May 2015	2015.05.04	Maintenance release.
Decmeber 2014	2014.12.15	Initial release.

Simulating the HPS Component 29

2016.10.28

a10_5v4 Subscribe Send Feedback

This section describes the simulation support for the hard processor system (HPS) component. The HPS simulation models provide bus functional models of the HPS and FPGA fabric and a simulation model of the interface to SDRAM memory.

The HPS simulation support does not include modules implemented in the HPS, such as the ARM Cortex-A9 MPCore processor.

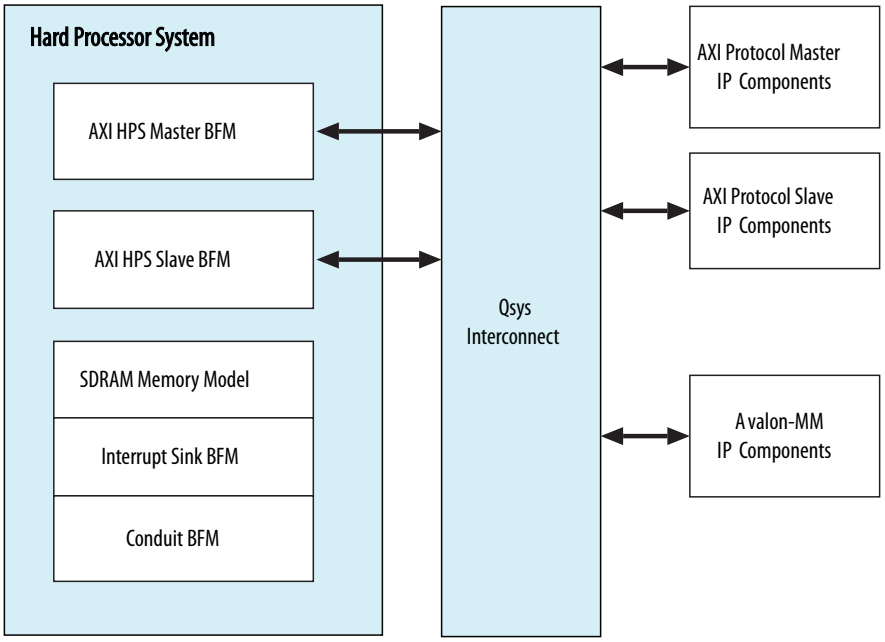
The simulation support files are specified when the HPS component is instantiated in the Qsys system integration tool. When you enable a particular HPS-FPGA interface, Qsys provides the corresponding model during the generation process.

The HPS simulation support enables you to develop and verify your own FPGA user logic or intellectual property (IP) that interfaces to the HPS component.

The simulation model supports the following interfaces:

- Clock and reset interfaces
- FPGA-to-HPS Advanced Microcontroller Bus Architecture (AMBA) Advanced eXtensible Interface (AXI) slave interface
- HPS-to-FPGA AXI master interface
- Lightweight HPS-to-FPGA AXI master interface
- Microprocessor unit (MPU) general-purpose I/O (GPIO) interface
- MPU standby and event interface
- Interrupts interface
- Direct memory access (DMA) controller peripheral request interface
- Debug Advanced Peripheral Bus (APB) interface
- System Trace Macrocell (STM) hardware event
- FPGA cross trigger interface (CTI)
- FPGA trace port interface unit (TPIU)
- Boot from FPGA interface

Figure 29-1: HPS BFM Block Diagram



The HPS BFMs use standard function calls from the Altera BFM application programming interface (API), as detailed in the remainder of this chapter.

HPS simulation supports only Verilog HDL or SystemVerilog simulation environments. Users with VHDL Custom IP can run BFM simulations so long as a mixed-language simulation license is available on their chosen simulator.

Related Information

- **Simulation Flows** on page 29-2
- **Instantiating the HPS Component** on page 27-1
- **Avalon Verification IP Suite User Guide**
- **Mentor Verification IP Altera Edition User Guide**

Simulation Flows

Altera provides a functional register transfer level (RTL) simulation and a post–fitter gate–level simulation flow. Both simulation flows involve the following major steps, which is defined in the following sections:

1. Setting up the HPS component for simulation.
2. Generating the HPS simulation model in Qsys.
3. Running the simulation.

Related Information

Simulating Altera Designs
For general information about simulation, refer to the *Simulating Altera Designs* chapter in volume 3 of the *Quartus Prime Handbook*.

Setting Up the HPS Component for Simulation

The following steps outline how to set up the HPS component for simulation.

1. Add the HPS component from the Qsys Component Library.
2. Configure the component based on your application needs by selecting or deselecting the HPS-FPGA interfaces.
3. Connect the appropriate HPS interfaces to other components in the system. For example, connect the FPGA-to-HPS AXI slave interface to an AXI or Avalon-MM master interface in another component in the system.

When you create your component, make sure the conduit interfaces have the correct role names and widths. Also make sure the conduit interfaces are opposite in direction to what is shown in the HPS Conduit Interfaces table.

Related Information

Instantiating the HPS Component on page 27-1

HPS Conduit Interfaces Connecting to the FPGA

The following tables define the HPS Conduit interfaces that connect to the FPGA.

Table 29-1: `h2f_warm_reset_handshake`

Role Name	Direction	Width
`h2f_pending_rst_req_n`	Output	1
`f2h_pending_rst_ack_n`	Input	1

Table 29-2: h2f_gp

Role Name	Direction	Width
`h2f_gp_in`	Input	32
`h2f_gp_out`	Output	32

Table 29-3: h2f_mpu_events

Role Name	Direction	Width
`h2f_mpu_eventi`	Input	1
`h2f_mpu_evento`	Output	1
`h2f_mpu_standbywfe`	Output	2
`h2f_mpu_standbywfi`	Output	2

Table 29-4: f2h_dma0 to f2h_dma7

Role Name	Direction	Width
f2h_dma_req<0-7>_req	Input	1
f2h_dma_req<0-7>_single	Input	1
f2h_dma_req<0-7>_ack	Output	1

Table 29-5: h2f_debug_apb_sideband

Role Name	Direction	Width
h2f_dbg_apb_PCLKEN	Input	1
h2f_dbg_apb_DBG_APB_DISABLE	Input	1

Table 29-6: f2h_stm_hw_events

Role Name	Direction	Width
f2h_stm_hwevents	Input	28

Table 29-7: h2f_cti

Role Name	Direction	Width
h2f_cti_trig_in	Input	8
h2f_cti_trig_in_ack	Output	8
h2f_cti_trig_out	Output	8
h2f_cti_trig_out_ack	Input	8
h2f_cti_fpga_clk_en	Input	1

Table 29-8: h2f_tpiu

Role Name	Direction	Width
h2f_tpiu_clk_ctrl	Input	1
h2f_tpiu_data	Output	32

Table 29-9: f2h_boot_from_fpga

Role Name	Direction	Width
f2h_boot_from_fpga_ready	Input	1
f2h_boot_from_fpga_on_failure	Input	1

Setting Up the HPS Component for Simulation

The following steps outline how to set up the HPS component for simulation.

1. Add the HPS component from the Qsys Component Library.
2. Configure the component based on your application needs by selecting or deselecting the HPS-FPGA interfaces.
3. Connect the appropriate HPS interfaces to other components in the system. For example, connect the FPGA-to-HPS AXI slave interface to an AXI or Avalon-MM master interface in another component in the system.

When you create your component, make sure the conduit interfaces have the correct role names and widths. Also make sure the conduit interfaces are opposite in direction to what is shown in the HPS Conduit Interfaces table.

Related Information

Instantiating the HPS Component on page 27-1

Generating the HPS Simulation Model in Qsys

The following steps outline how to generate the simulation model:

1. In Qsys, click **Generate HDL** under the Generate menu.
2. Choose between RTL and post–fit simulation

   For RTL simulation, perform the following steps:

   a. Set **Create simulation model** to Verilog.
   b. Click **Generate**.[58]

   For post–fit simulation, perform the following steps:

   a. Turn on the **Create HDL design files for synthesis** option.
   b. Turn on the **Create block symbol file (.bsf)** option.[59][60]
3. Click **Generate**.

Related Information

- **Instantiating the HPS Component** on page 27-1
- **Creating a System with Qsys**
  For more information about Qsys simulation, refer to "Simulating a Qsys System" in the *Creating a System with Qsys* chapter in volume 1 of the *Quartus Prime Handbook*.

Running the Simulations

The steps to run a simulation depend on whether you are running an RTL simulation or a post-fit simulation.

[58] VHDL is supported for HPS simulation and it requires a mix language simulator. However, the BFMs always need to be in verilog. Custom components can be in VHDL.
[59] A **.bsf** file is only needed for schematic entry.
[60] This is not a requirement for simulation or implementation unless a schematic is used.

Running HPS RTL Simulation

Qsys generates scripts for several simulators that you can use to complete the simulation process, as listed in the following table.

Table 29-10: Qsys-Generated Scripts for Supported Simulators

Simulator	Script Name	Directory
Mentor Graphics® Modelsim Altera Edition	**msim_setup.tcl**	*<project directory>*/*<Qsys design name>*/**simulation/mentor**
Cadence NC-Sim	**ncsim_setup.sh**	*<project directory>*/*<Qsys design name>*/**simulation/cadence**
Synopsys VCS	**vcs_setup.sh**	*<project directory>*/*<Qsys design name>*/**simulation/synopsys/vcs**
Synopsys VCS-MX	**vcsmx_setup.sh**	*<project directory>*/*<Qsys design name>*/**simulation/synopsys/vcsmx**
Aldec® RivieraPro™	**rivierapro_setup.tcl**	*<project directory>*/*<Qsys design name>*/**simulation/aldec**

Related Information

- **Avalon Verification IP Suite User Guide**
- **Mentor Verification IP Altera Edition User Guide**

Running HPS Post-Fit Simulation

To run HPS post-fit simulation after successful Qsys generation, perform the following steps:

1. Add the generated synthesis file set to your Quartus Prime project by performing the following steps:
   a. In the Quartus Prime software, click **Settings** in the Assignments menu.
   b. In the **Settings** *<your Qsys system name>* dialog box, on the **Files** tab, browse to *<your project directory>*/*<your Qsys system name>*/**synthesis/** and select *<your Qsys system name>*.**qip**.
   c. Click **Open**. The **Select File** dialog box closes.
   d. Click **OK**. The **Settings** dialog box closes.
2. Optionally instantiate your HPS system as the top-level entity in your Quartus Prime project.
3. Compile the design by clicking **Start Compilation** in the Processing menu.
4. Change the EDA Netlist Writer settings, if necessary, by performing the following steps:
   a. Click **Settings** in the Assignment menu.
   b. On the **Simulation** tab, under the **EDA Tool Settings** tab, you can specify the following EDA Netlist Writer settings:

- **Tool name**—The name of the simulation tool
- **Format for output netlist**
- **Output directory**

c. Click **OK**.

5. To create the post-fitter simulation model with Quartus Prime EDA Netlist Writer, perform the following steps:

a. Click **Start** in the Processing menu.
b. Click **Start EDA Netlist Writer**.

Related Information

Quartus Prime Help

Post-Fit Simulation Files

Post-fit simulation is the simulation of the netlist generated from the original RTL design after it has been mapped, synthesized, and fit. The netlist represents the actual hardware and its connections as they appear in the FPGA. Quartus Prime generates the netlist and can generate a Standard Delay Format (**.sdf**) file with the timing information for all connections. The simulation can be functional only (without the timing information) where all wires and gates take zero time, or it can be a timing simulation where the time for all transitions is based on the SDF information.

Post-fit simulation can serve a number of different purposes. It can be used to perform a dynamic verification of the timing of the design, or it can be used to verify the functional correctness of either the design, the compilation flow (in particular, the fitter), or both.

This table uses the following symbols:

- *<ACDS install>* = Altera Complete Design Suite installation path
- *<Avalon Verification IP>* = *<ACDS install>***/ip/altera/sopc_builder_ip/verification**
- *<AXI Verification IP>* = *<ACDS install>***/ip/altera/mentor_vip_ae**
- *<HPS Post-fit Sim>* = *<ACDS install>***/ip/altera/hps/postfitter_simulation**
- *<Device Sim Lib>* = *<ACDS install>***/quartus/eda/sim_lib**

Table 29-11: Post-Fit Simulation Files

Library	Directory	File
Altera Verification IP Library	*<Avalon Verification IP>***/lib/**	**verbosity_pkg.sv** **avalon_mm_pkg.sv** **avalon_utilities_pkg.sv**
Avalon Clock Source BFM	*<Avalon Verification IP>*/ **altera_avalon_clock_source/**	**altera_avalon_clock_source.sv**
Avalon Reset Source BFM	*<Avalon Verification IP>*/ **altera_avalon_reset_source/**	**altera_avalon_reset_source.sv**
Avalon MM Slave BFM	*<Avalon Verification IP>*/ **altera_avalon_mm_slave_bfm/**	**altera_avalon_mm_slave_bfm.sv**

Library	Directory	File
Avalon Interrupt Sink BFM	*<Avalon Verification IP>*/**altera_avalon_interrupt_sink/**	**altera_avalon_interrupt_sink.sv**
Mentor AXI Verification IP Library	*<AXI Verification IP>*/**common/**	**questa_mvc_svapi.svh**
Mentor AXI3 BFM	*<AXI Verification IP>*/**axi3/axi3/bfm/**	**mgc_common_axi.sv** **mgc_axi_master.sv** **mgc_axi_slave.sv**
HPS Post-Fit Simulation Library	*<HPS Post-fit Sim>*/	**All the files in the directory**
Device Simulation Library(61)	*<Device Sim Lib>*/	**altera_primitives.v** **220model.v** **sgate.v** **altera_mf.v** **altera_lnsim.sv** **cyclonev_atoms.v** **arriav_atoms.v** **mentor/cyclonev_atoms_ncrypt.v** **mentor/arriav_atoms_ncrypt.v**
EDA Netlist Writer Generated Post-Fit Simulation Model	*<User project directory>*/	***.vo** ***.vho** (Mixed-language simulator is needed for Verilog HDL and VHDL mixed design)
User testbench files	*<User project directory>*/	***.v** ***.sv** ***.vhd** (Mixed-language simulator is needed for Verilog HDL and VHDL mixed design)

(61) The device simulation library is not needed with Modelsim-Altera.

BFM API Hierarchy Format

For post-fit simulation, you must call the BFM API in your test program with a specific hierarchy. The hierarchy format is:

```
<DUT>.\<HPS>|fpga_interfaces|<interface><space>.<BFM>.<API function>
```

Where:

- <*DUT*> is the instance name of the design under test that you instantiated in your test bench . The design under test is the HPS component.
- <*HPS*> is the HPS component instance name that you use in your Qsys system.
- <*interface*> is the instance name of a specific FPGA-to-HPS or HPS-to-FPGA interface. This name can be found in the **fpga_interfaces.sv** file located in <*project directory*>/<*Qsys design name*>/**synthesis/submodules**.
- <*space*>—You must insert one space character after the interface instance name.
- <*BFM*> is the BFM instance name. To identify the BFM instance name, in <*ACDS install*>/**ip/altera/hps/postfitter_simulation**, find the SystemVerilog file corresponding to the interface type that you are using. This SystemVerilog file contains the BFM instance name.

For example, a path for the Lightweight HPS-to-FPGA master interface hierarchy could be formed as follows:

```
top.dut.\my_hps_component|fpga_interface|hps2fpga_light_weight .h2f_lw_axi_master
```

Notice the space after `hps2fpga_light_weight`. Omitting this space would cause simulation failure because the instance name `hps2fpga_light_weight` , including the space, is the name used in the post-fit simulation model generated by the Quartus Prime software.

Clock and Reset Interfaces

Clock Interface

Qsys generates the clock source BFM for the FPGA-to-HPS alternate clock source.

Table 29-12: HPS Clock Input Interface Simulation Model

The Altera clock source BFM application programming interface (API) applies to the BFM listed in this table. Your Verilog interfaces use the same API.

Interface Name	BFM Instance Name
`f2h_free_clk`	`f2h_free_clock_inst`

Qsys generates the clock source BFM for each clock output interface from the HPS component. For HPS-to-FPGA user clocks, specify the BFM clock rate in the **User clock frequency field** in the **HPS Clocks** page when instantiating the HPS component in Qsys.

The HPS-to-FPGA debug APB interface generates a clock output to the FPGA, named `h2f_debug_apb_clock`. In simulation, the clock source BFM also represents this clock output's behavior.

Table 29-13: HPS Clock Output Interface Simulation Model

The Altera clock source BFM application programming interface (API) applies to all the BFMs listed in this table. Your Verilog interfaces use the same API.

Interface Name	BFM Instance Name
h2f_user0_clock	h2f_user0_clock_inst
h2f_user1_clock	h2f_user1_clock_inst
h2f_user2_clock	h2f_user2_clock_inst

Related Information

Memory-Mapped Interfaces on page 28-1

Reset Interface

The HPS reset request and handshake interfaces are connected to Altera conduit BFMs for simulation.

Table 29-14: HPS Reset Input Interface Simulation Model

You can monitor the reset request interface state changes or set the interface by using the API listed.

Interface Name	BFM Instance Name	API Function Names
f2h_cold_reset_req	f2h_cold_reset_req_inst	get_f2h_cold_rst_req_n()
f2h_debug_reset_req	f2h_debug_reset_req_inst	get_f2h_dbg_rst_req_n()
f2h_warm_reset_req	f2h_warm_reset_req_inst	get_f2h_warm_rst_req_n()
h2f_warm_reset_handshake	h2f_warm_reset_handshake_inst	set_h2f_pending_rst_req_n() get_f2h_pending_rst_ack_n()

Table 29-15: HPS Reset Output Interface Simulation Model

The Altera reset source BFM application programming interface applies to all the BFMs listed.

Interface Name	BFM Instance Name
h2f_reset	h2f_reset_inst
h2f_cold_reset	h2f_cold_reset_inst
h2f_debug_apb_reset	h2f_debug_apb_reset_inst

Table 29-16: Configuration of Reset Source BFM for HPS Reset Output Interface

The HPS reset output interface is connected to a reset source BFM. Qsys configures the BFM as shown in the following table. The parameter value of the instantiated BFM is configured for HPS simulation.

Parameter	BFM Value	Meaning
Assert reset high	Off	This parameter is off, specifying an active-low reset signal from the BFM.

Parameter	BFM Value	Meaning
Cycles of initial reset	0	This parameter is 0, specifying that the BFM does not assert the reset signal automatically.

Related Information

- **Memory-Mapped Interfaces** on page 28-1
- **Avalon Verification IP Suite User Guide**

FPGA-to-HPS AXI Slave Interface

The FPGA-to-HPS AXI slave interface, `f2h_axi_slave`, is connected to a Mentor Graphics AXI slave BFM for simulation with an instance name of `f2h_axi_slave_inst`. Qsys configures the BFM as shown in the following table. The BFM clock input is connected to `f2h_axi_clock` clock.

Table 29-17: Configuration of FPGA-to-HPS AXI Slave BFM

Parameter	Value
AXI Address Width	32
AXI Read Data Width	32, 64, or 128
AXI Write Data Width	32, 64, or 128
AXI ID Width	8

You control and monitor the AXI slave BFM by using the BFM API.

Related Information

- **Memory-Mapped Interfaces** on page 28-1
- **Mentor Verification IP Altera Edition User Guide**
  The Mentor Verification IP User guide provides details of the API and connection guidelines for the AXI3 and AXI4 BFMs.

HPS-to-FPGA AXI Master Interface

The HPS-to-FPGA AXI master interface, `h2f_axi_master`, is connected to a Mentor Graphics AXI master BFM for simulation with an instance name of `h2f_axi_master_inst`. In Qsys, you can configure the HPS-to-FPGA interface with the following address, data, and ID widths. The BFM clock input is connected to `h2f_axi_clock` clock.

Table 29-18: Configuration of HPS-to-FPGA AXI Master BFM

Parameter	Value
AXI Address Width	30
AXI Read and Write Data Width	32, 64, or 128
AXI ID Width	12

You control and monitor the AXI master BFM by using the BFM API.

Related Information

- **Memory-Mapped Interfaces** on page 28-1
- **Mentor Verification IP Altera Edition User Guide**
  The Mentor Verification IP User guide provides details of the API and connection guidelines for the AXI3 and AXI4 BFMs.

Lightweight HPS-to-FPGA AXI Master Interface

The lightweight HPS-to-FPGA AXI master interface, `h2f_lw_axi_master`, is connected to a Mentor Graphics AXI master BFM for simulation with an instance name of `h2f_lw_axi_master_inst`. Qsys configures the BFM as shown in the following table. The BFM clock input is connected to `h2f_lw_axi_clock` clock.

Table 29-19: Configuration of Lightweight HPS-to-FPGA AXI Master BFM

Parameter	Value
AXI Address Width	21
AXI Read and Write Data Width	32
AXI ID Width	12

You control and monitor the AXI master BFM by using the BFM API.

Related Information

- **Memory-Mapped Interfaces** on page 28-1
- **Mentor Verification IP Altera Edition User Guide**
  The Mentor Verification IP User guide provides details of the API and connection guidelines for the AXI3 and AXI4 BFMs.

HPS-to-FPGA MPU Event Interface

The HPS-to-FPGA MPU event interface is connected to an Altera conduit BFM for simulation. The following table lists the name of each interface, along with API function names for each type of simulation. You can monitor the interface state changes or set the interface by using the API listed.

Table 29-20: HPS-to-FPGA MPU Event Interface Simulation Model

The usage of conduit `get_*()` and `set_*()` API functions is the same as with the general Avalon conduit BFM.

Interface Name	BFM Instance Name	RTL Simulation API Function Names	Post-Fit Simulation API Function Names

`h2f_mpu_events`	`h2f_mpu_events_inst`	`get_h2f_mpu_eventi()`
		`set_h2f_mpu_evento()`
		`set_h2f_mpu_standbywfe()`
		`set_h2f_mpu_standbywfi()`
			`get_eventi()`
			`set_evento()`
			`set_standbywfe()`
			`set_standbywfi()`

Related Information

Avalon Verification IP Suite User Guide

Interrupts Interface

The FPGA-to-HPS interrupts interface is connected to an Altera Avalon interrupt sink BFM for simulation.

Table 29-21: FPGA-to-HPS Interrupts Interface Simulation Model

Interface Name	BFM Instance Name
`f2h_irq0`	`f2h_irq0_inst`
`f2h_irq1`	`f2h_irq1_inst`

The HPS-to-FPGA peripheral interfaces are connected to Altera conduit BFMs for simulation. When you enable the peripheral interrupt, the corresponding peripheral signal to the FPGA is exposed.

Table 29-22: HPS-to-FPGA Peripherals Interrupt Interface Simulation Model

Interface Name	BFM Instance Name
`h2f_clkmgr_interrupt`	`h2f_clkmgr_interrupt_inst`
`h2f_mpuwakeup_interrupt`	`h2f_mpuwakeup_interrupt_inst`

Interface Name	BFM Instance Name
h2f_ncti_interrupt0	h2f_ncti_interrupt0_inst
h2f_ncti_interrupt1	h2f_ncti_interrupt1_inst
h2f_dma_interrupt0	h2f_dma_interrupt0_inst
h2f_dma_interrupt1	h2f_dma_interrupt1_inst
h2f_dma_interrupt2	h2f_dma_interrupt2_inst
h2f_dma_interrupt3	h2f_dma_interrupt3_inst
h2f_dma_interrupt4	h2f_dma_interrupt4_inst
h2f_dma_interrupt5	h2f_dma_interrupt5_inst
h2f_dma_interrupt6	h2f_dma_interrupt6_inst
h2f_dma_interrupt7	h2f_dma_interrupt7_inst
h2f_dma_abort_interrupt	h2f_dma_abort_interrupt_inst
h2f_fpga_man_interrupt	h2f_fpga_man_interrupt_inst
h2f_gpio0_interrupt	h2f_gpio0_interrupt_inst
h2f_gpio1_interrupt	h2f_gpio1_interrupt_inst
h2f_gpio2_interrupt	h2f_gpio2_interrupt_inst
h2f_hmc_interrupt	h2f_hmc_interrupt_inst
h2f_timer_l4sp_0_interrupt	h2f_timer_l4sp_0_interrupt_inst
h2f_timer_l4sp_1_interrupt	h2f_timer_l4sp_1_interrupt_inst
h2f_timer_sys_0_interrupt	h2f_timer_sys_0_interrupt_inst
h2f_timer_sys_1_interrupt	h2f_timer_sys_1_interrupt_inst
h2f_ecc_serr_interrupt	h2f_ecc_serr_interrupt_inst
h2f_ecc_derr_interrupt	h2f_ecc_derr_interrupt_inst
h2f_parity_l1_interrupt	h2f_parity_l1_interrupt_inst
h2f_wdog0_interrupt	h2f_wdog0_interrupt_inst
h2f_wdog1_interrupt	h2f_wdog1_interrupt_inst

HPS-to-FPGA Debug APB Interface

The HPS-to-FPGA debug APB interface is connected to an Altera conduit BFM for simulation. The following table lists the name of each interface, along with API function names for each type of simulation. You can monitor the interface state changes or set the interface by using the API functions listed.

Table 29-23: HPS-to-FPGA Debug APB Interface Simulation Model

Interface Name	BFM Name	RTL Simulation API Function Names	Post-Fit Simulation API Function Names
h2f_debug_apb	h2f_debug_apb	set_h2f_dbg_apb_PADDR() set_h2f_dbg_apb_PADDR_31() set_h2f_dbg_apb_PENABLE() get_h2f_dbg_apb_PRDATA() get_h2f_dbg_apb_PREADY() set_h2f_dbg_apb_PSEL() get_h2f_dbg_apb_PSLVERR() set_h2f_dbg_apb_PWDATA() set_h2f_dbg_apb_PWRITE()	set_PADDR() set_PADDR_31() set_PENABLE() get_PRDATA() get_PREADY() set_PSEL() get_PSLVERR() set_PWDATA() set_PWRITE()
h2f_debug_apb_sideband	h2f_debug_apb_sideband	get_h2f_dbg_apb_PCLKEN() get_h2f_dbg_apb_DBG_APB_DISABLE()	get_PCLKEN() get_DBG_APB_DISABLE()

FPGA-to-HPS System Trace Macrocell Hardware Event Interface

The FPGA-to-HPS STM hardware event interface is connected to an Altera conduit BFM for simulation. The following table lists the name of each interface, along with the API function name for each type of simulation. You can monitor the interface state changes or set the interface by using the API functions listed.

Table 29-24: FPGA-to-HPS STM Hardware Event Interface Simulation Model

Interface Name	BFM Name	RTL Simulation API Function Name	Post-Fit Simulation API Function Name
f2h_stm_hw_events	f2h_stm_hw_events_inst	get_f2h_stm_hwevents()	get_stm_events()

HPS-to-FPGA Cross-Trigger Interface

The HPS-to-FPGA cross-trigger interface is connected to an Altera conduit BFM for simulation. The following table lists the name of each interface, along with API function names for each type of simulation. You can monitor the interface state changes or set the interface by using the API functions listed.

Table 29-25: HPS-to-FPGA Cross-Trigger Interface Simulation Model

Interface Name	BFM Name	RTL Simulation API Function Names	Post-Fit Simulation API Function Names
h2f_cti	h2f_cti_inst	get_h2f_cti_trig_in()	get_trig_in()
		set_h2f_cti_trig_in_ack()	set_trig_inack()
		set_h2f_cti_trig_out()	set_trig_out()
		get_h2f_cti_trig_out_ack()	get_trig_outack()
		get_h2f_cti_fpga_clk_en()	get_clk_en()

FPGA-to-HPS DMA Handshake Interface

The FPGA-to-HPS DMA handshake interface is connected to an Altera conduit BFM for simulation. The following table lists the name for each interface, along with API function names for each type of simulation. You can monitor the interface state changes or set the interface by using the API listed.

Table 29-26: FPGA-to-HPS DMA Handshake Interface Simulation Model

The usage of conduit `get_*()` and `set_*()` API functions is the same as with the general Avalon conduit BFM.

Interface Name	BFM Instance Name	RTL Simulation API Function Names	Post-Fit Simulation API Function Names
f2h_dma0	f2h_dma0_inst	get_f2h_dma0_req() get_f2h_dma0_single() set_f2h_dma0_ack()	get_channel0_req() get_channel0_single() set_channel0_xx_ack()
f2h_dma1	f2h_dma1_inst	get_f2h_dma1_req() get_f2h_dma1_single() set_f2h_dma1_ack()	get_channel1_req() get_channel1_single() set_channel1_xx_ack()

Interface Name	BFM Instance Name	RTL Simulation API Function Names	Post-Fit Simulation API Function Names
f2h_dma2	f2h_dma2_inst	get_f2h_dma2_req() get_f2h_dma2_single() set_f2h_dma2_ack()	get_channel2_req() get_channel2_single() set_channel2_xx_ack()
f2h_dma3	f2h_dma3_inst	get_f2h_dma3_req() get_f2h_dma3_single() set_f2h_dma3_ack()	get_channel3_req() get_channel3_single() set_channel3_xx_ack()
f2h_dma4	f2h_dma4_inst	get_f2h_dma4_req() get_f2h_dma4_single() set_f2h_dma4_ack()	get_channel4_req() get_channel4_single() set_channel4_xx_ack()
f2h_dma5	f2h_dma5_inst	get_f2h_dma5_req() get_f2h_dma5_single() set_f2h_dma5_ack()	get_channel5_req() get_channel5_single() set_channel5_xx_ack()
f2h_dma6	f2h_dma6_inst	get_f2h_dma6_req() get_f2h_dma6_single() set_f2h_dma6_ack()	get_channel6_req() get_channel6_single() set_channel6_xx_ack()
f2h_dma7	f2h_dma7_inst	get_f2h_dma7_req() get_f2h_dma7_single() set_f2h_dma7_ack()	get_channel7_req() get_channel7_single() set_channel7_xx_ack()

Related Information

Avalon Verification IP Suite User Guide

Boot from FPGA Interface

The boot from FPGA interface is connected to an Altera conduit BFM for simulation. You can monitor the interface state changes or set the interface by using the API functions in the table below.

Table 29-27: Boot from FPGA Interface Simulation Model

Interface Name	BFM Name	RTL simulation API Function Names	Post-fit Simulation API Function Names
f2h_boot_from_fpga	f2h_boot_from_fpga_inst	get_f2h_boot_from_fpga_ready() get_f2h_boot_from_fpga_on_failure()	get_boot_fpga_ready() get_boot_from_fpga_on_failure()

Security Manager Anti-Tamper Signals Interface

The security manager anti-tamper signals interface is connected to an Altera conduit BFM for simulation. You can monitor the interface state changes or set the interface by using the API listed.

Table 29-28: Security Manager Anti-Tamper Signals Interface Simulation Model

Interface Name	BFM Instance Name	RTL Simulation API Function Names
h2f_security	h2f_security_inst	get_f2h_security_anti_tamper_in set_h2f_security_anti_tamper_out

EMIF Conduit

Enables the HPS dedicated conduit to the Arria 10 External memory Interface for HPS. This conduit cannot connect to any other External memory Interface (EMIF). Only IP generated by the Arria 10 External memory Interface for HPS QSYS library should be used.

Table 29-29: EMIF Conduit Interface Simulation Model

Interface Name	BFM Instance Name	RTL Simulation API Function Names
emif	emif_inst	emif_emif_to_hps emif_hps_to_emif

Pin MUX and Peripherals

Under Pin MUX and Peripherals, the **Pin Mux GUI** has three folder tabs with the following functions:

- IP Selection
- Advanced Pin Placement
- Advanced FPGA Placement

The **IP Selection** tab contains two sub-windows. In the left sub-window, you are able to select the following:

- Boot source — SD/MMC, NAND, and QSPI
- Pin routing decisions between:
  - IP and HPS I/O—where your selections are reflected on the **Advanced Pin Placement**
  - IP and FPGA—where your selections are reflected on the **Advanced FPGA Placement**
- NAND bit-width when the NAND IP is selected
- SD/MMC bit-width and SD/MMC Power Enable when the SD/MMC IP is selected
- RGMII and PHY Options for each of the EMACs when the EMAC IP is selected
- Additional QSPI Slave Selects when the QSPI IP is selected

After you have made your selection, you must click on the **Apply Selections** button and select "Show signals" so that you can see your selection reflected in the right sub-window. Depending on if you are routing to the HPS I/O or the FPGA, your selection is reflected on the **Advanced Pin Placement** or **Advanced FPGA Placement** tab, respectively.

Related Information

Configuring Peripherals on page 27-14

HPS Conduit Interfaces Connecting to the HPS I/O

The **Pin Mux and Peripherals** interface, `hps_io`, is connected to an Altera conduit BFM for simulation. Qsys configures the BFM as shown in the following tables:

Table 29-30: PLL Pins Interface Simulation Model

To HPS I/Os	Options	Signal(s)
1 .. 5	N/A	`hps_io_phery_cm_PLL_CLK0 ()` .. `hps_io_phery_cm_PLL_CLK4 ()`

Table 29-31: SD/MMC Interface Simulation Model

This peripheral is a boot source.

To HPS I/Os	Options	Signal(s)

1	**SDMMC bit-width** = Default (1)
	SDMMC Power Enable = Yes
		`hps_io_phery_sdmmc_CMD()`
		`hps_io_phery_sdmmc_PWR_ENA()`, only when **SDMMC Power Enable** option is set to "Yes"
		`hps_io_phery_sdmmc_D0()`
		`hps_io_phery_sdmmc_CCLK ()`

	SDMMC bit-width = 4
	SDMMC Power Enable = Yes(62)
		The same signal list from above with **SDMMC bit-width** set to "default", plus the following signals:
		`hps_io_phery_sdmmc_D1()`
		`hps_io_phery_sdmmc_D2()`
		`hps_io_phery_sdmmc_D3()`

	SDMMC bit-width = 8
	SDMMC Power Enable = Yes(62)
		The same signal list from above with **SDMMC bit-width** set to 4, plus the following signals:
		`hps_io_phery_sdmmc_D4()`
		`hps_io_phery_sdmmc_D5()`
		`hps_io_phery_sdmmc_D6()`
		`hps_io_phery_sdmmc_D7()`

(62) When **SDMMC Power Enable** is set to Default, the `hps_io_phery_sdmmc_PWR_ENA()` signal is not available.

Table 29-32: USB Pins Interface Simulation Model

To HPS I/Os	Options	Signal(s)

1		`hps_io_phery_usb1_DATA0 ()`
		`hps_io_phery_usb1_DATA1 ()`
		`hps_io_phery_usb1_DATA2 ()`
		`hps_io_phery_usb1_DATA3 ()`
		`hps_io_phery_usb1_DATA4 ()`
		`hps_io_phery_usb1_DATA5 ()`
		`hps_io_phery_usb1_DATA6 ()`
		`hps_io_phery_usb1_DATA7 ()`
		`hps_io_phery_usb1_CLK ()`
		`hps_io_phery_usb1_STP ()`
		`hps_io_phery_usb1_DIR ()`
		`hps_io_phery_usb1_NXT ()`

2	N/A	`hps_io_phery_usb0_DATA0 ()`
		`hps_io_phery_usb0_DATA1 ()`
		`hps_io_phery_usb0_DATA2 ()`
		`hps_io_phery_usb0_DATA3 ()`
		`hps_io_phery_usb0_DATA4 ()`
		`hps_io_phery_usb0_DATA5 ()`
		`hps_io_phery_usb0_DATA6 ()`
		`hps_io_phery_usb0_DATA7 ()`
		`hps_io_phery_usb0_CLK ()`
		`hps_io_phery_usb0_STP ()`
		`hps_io_phery_usb0_DIR ()`
		`hps_io_phery_usb0_NXT ()`
		Add the same signal list from above to this list of signals.

Table 29-33: EMAC Pins Interface Simulation Model

To HPS I/Os	Options	Signal(s)

1	EMAC A is available for selecting options. See the signals in the "Signal(s)" column.	**RGMII?** = RMII and **PHY Options** = None (Default List)
		Note: Where <i> is "To HPS I/Os" - 1.
		`hps_io_phery_emac<i>_TX_CLK ()`
		`hps_io_phery_emac<i>_TXD0 ()`

To HPS I/Os	Options	Signal(s)
		hps_io_phery_emac<i>_TXD1 () hps_io_phery_emac<i>_RX_CTL () hps_io_phery_emac<i>_TX_CTL () hps_io_phery_emac<i>_RX_CLK () hps_io_phery_emac<i>_RXD0 () hps_io_phery_emac<i>_RXD1 () When **RGMII?** = RGMII Add the following signals to the Default List: hps_io_phery_emac<i>_TXD2 () hps_io_phery_emac<i>_TXD3 () hps_io_phery_emac<i>_RXD2 () hps_io_phery_emac<i>_RXD3 () When **PHY Options** = MDIO Add the following signals to the Default List: hps_io_phery_emac<i>_MDIO () hps_io_phery_emac<i>_MDC () When **PHY Options** = I2C Add the following signals to the Default List: hps_io_phery_i2cemac<i>_SDA () hps_io_phery_i2cemac<i>_SCL ()

To HPS I/Os	Options	Signal(s)

2	EMAC A and EMAC B are available for selecting options. See the signals in the "Signal(s)" column.	This signal list is comprised of the list for when "To HPS I/Os" is equal to 1, taking in to consideration the various options, plus the following:
		Note: Where <i> is "To HPS I/Os" - 1.
		RGMII? = RMII and **PHY Options** = None (Default List)
		`hps_io_phery_emac<i>_TX_CLK ()`
		`hps_io_phery_emac<i>_TXD0 ()`
		`hps_io_phery_emac<i>_TXD1 ()`
		`hps_io_phery_emac<i>_RX_CTL ()`
		`hps_io_phery_emac<i>_TX_CTL ()`
		`hps_io_phery_emac<i>_RX_CLK ()`
		`hps_io_phery_emac<i>_RXD0 ()`
		`hps_io_phery_emac<i>_RXD1 ()`
		When **RGMII?** = RGMII
		Add the following signals to the Default List:
		`hps_io_phery_emac<i>_TXD2 ()`
		`hps_io_phery_emac<i>_TXD3 ()`
		`hps_io_phery_emac<i>_RXD2 ()`
		`hps_io_phery_emac<i>_RXD3 ()`
		When **PHY Options** = MDIO
		Add the following signals to the Default List:
		`hps_io_phery_emac<i>_MDIO ()`
		`hps_io_phery_emac<i>_MDC ()`
		When **PHY Options** = I2C
		Add the following signals to the Default List:
		`hps_io_phery_i2cemac<i>_SDA ()`
		`hps_io_phery_i2cemac<i>_SCL ()`

To HPS I/Os	Options	Signal(s)

3	EMAC A, EMAC B, and EMAC C are available for selecting options. See the signals in the "Signal(s)" column.	This signal list is comprised of the list for when "To HPS I/Os" is equal to 1 and 2, taking in to consideration the various options, plus the following:
		Note: Where <i> is "To HPS I/Os" - 1.
		RGMII? = RMII and **PHY Options** = None (Default List)
		`hps_io_phery_emac<i>_TX_CLK ()`
		`hps_io_phery_emac<i>_TXD0 ()`
		`hps_io_phery_emac<i>_TXD1 ()`
		`hps_io_phery_emac<i>_RX_CTL ()`
		`hps_io_phery_emac<i>_TX_CTL ()`
		`hps_io_phery_emac<i>_RX_CLK ()`
		`hps_io_phery_emac<i>_RXD0 ()`
		`hps_io_phery_emac<i>_RXD1 ()`
		When **RGMII?** = RGMII
		Add the following signals to the Default List:
		`hps_io_phery_emac<i>_TXD2 ()`
		`hps_io_phery_emac<i>_TXD3 ()`
		`hps_io_phery_emac<i>_RXD2 ()`
		`hps_io_phery_emac<i>_RXD3 ()`
		When **PHY Options** = MDIO
		Add the following signals to the Default List:
		`hps_io_phery_emac<i>_MDIO ()`
		`hps_io_phery_emac<i>_MDC ()`
		When **PHY Options** = I2C
		Add the following signals to the Default List:
		`hps_io_phery_i2cemac<i>_SDA ()`
		`hps_io_phery_i2cemac<i>_SCL ()`

Table 29-34: SPIM Pins Interface Simulation Model

To HPS I/Os	Options	Signal(s)

1	N/A	`hps_io_phery_spim0_CLK ()`
		`hps_io_phery_spim0_MOSI ()`
		`hps_io_phery_spim0_MISO ()`
		`hps_io_phery_spim0_SS0_N ()`

2		The same list as above plus the following signals:
		`hps_io_phery_spim1_CLK ()`
		`hps_io_phery_spim1_MOSI ()`
		`hps_io_phery_spim1_MISO ()`
		`hps_io_phery_spim1_SS0_N ()`

Table 29-35: SPIS Pins Interface Simulation Model

To HPS I/Os	Options	Signal(s)

1	N/A	`hps_io_phery_spis0_CLK ()`
		`hps_io_phery_spis0_MOSI ()`
		`hps_io_phery_spis0_MISO ()`
		`hps_io_phery_spis0_SS0_N ()`

2		The same list as above plus the following signals:
		`hps_io_phery_spis1_CLK ()`
		`hps_io_phery_spis1_MOSI ()`
		`hps_io_phery_spis1_MISO ()`
		`hps_io_phery_spis1_SS0_N ()`

Table 29-36: UART Pins Interface Simulation Model

To HPS I/Os	Options	Signal(s)

1	N/A	`hps_io_phery_uart0_RX ()`
		`hps_io_phery_uart0_TX ()`
		`hps_io_phery_uart0_CTS_N ()`
		`hps_io_phery_uart0_RTS_N ()`

2		The same list as above plus the following signals:
		`hps_io_phery_uart1_RX ()`
		`hps_io_phery_uart1_TX ()`
		`hps_io_phery_uart1_CTS_N ()`
		`hps_io_phery_uart1_RTS_N ()`

Table 29-37: I²C Pins Interface Simulation Model

To HPS I/Os	Options	Signal(s)
1	N/A	`hps_io_phery_i2c0_SDA ()` `hps_io_phery_i2c0_SCL ()`

2		The same list as above plus the following signals:
		`hps_io_phery_i2c1_SDA ()`
		`hps_io_phery_i2c1_SCL ()`

Table 29-38: NAND Interface Simulation Model

This peripheral is a boot source.

To HPS I/Os	Options	Signal(s)
1	**NAND bit-width** = 8	hps_io_phery_nand_ALE () hps_io_phery_nand_CE_N () hps_io_phery_nand_CLE () hps_io_phery_nand_RE_N () hps_io_phery_nand_RB () hps_io_phery_nand_ADQ0 () hps_io_phery_nand_ADQ1 () hps_io_phery_nand_ADQ2 () hps_io_phery_nand_ADQ3 () hps_io_phery_nand_ADQ4 () hps_io_phery_nand_ADQ5 () hps_io_phery_nand_ADQ6 () hps_io_phery_nand_ADQ7 () hps_io_phery_nand_WP_N () hps_io_phery_nand_WE_N ()
	NAND bit-width = 16	The same list as above plus the following signals: hps_io_phery_nand_ADQ8 () hps_io_phery_nand_ADQ9 () hps_io_phery_nand_ADQ10 () hps_io_phery_nand_ADQ11 () hps_io_phery_nand_ADQ12 () hps_io_phery_nand_ADQ13 () hps_io_phery_nand_ADQ14 () hps_io_phery_nand_ADQ15 ()

Table 29-39: TRACE Pins Interface Simulation Model

To HPS I/Os	Options	Signal(s)
1	N/A	hps_io_phery_trace_CLK () hps_io_phery_trace_D0 () hps_io_phery_trace_D1 () hps_io_phery_trace_D2 () hps_io_phery_trace_D3 ()

Table 29-40: GPIO Pins Interface Simulation Model

To HPS I/Os	Options	Signal(s)
1 .. 62	N/A	`hps_io_gpio_gpio2_io0 ()` .. `hps_io_gpio_gpio2_io62 ()`

Table 29-41: QSPI Interface Simulation Model

This peripheral is a boot source.

To HPS I/Os	Options	Signal(s)

1	N/A
	Note: Although there are no options for this peripheral, "Boot" must be selected also for the signals to become available.
		`hps_io_phery_qspi_IO0()`
		`hps_io_phery_qspi_IO1()`
		`hps_io_phery_qspi_IO2_WPN()`
		`hps_io_phery_qspi_IO3_HOLD()`
		`hps_io_phery_qspi_CLK ()`
		`hps_io_phery_qspi_SS0()`

HPS Conduit Interfaces Connecting to the FPGA

When connecting to the FPGA, the **Pin Mux and Peripherals** interface is indicative to the peripheral name and represented in the "Conduit" column in the following table. The following tables for each of the peripherals, lists the conduits and their interfaces. In order to see the signals, you must select "Show Signals".

Table 29-42: SD/MMC Interface Simulation Model

To FPGA	Conduit	Interface(s)
1	`sdmmc`	`sdmmc_reset` `sdmmc_clk`

Table 29-43: EMAC Pins Interface Simulation Model

To FPGA	Conduit	Interface(s)
1	emac0 i2cemac0 — when **PHY Options** is set to "I2C"	emac_ptp_ref_clock emac0_rx_clk_in emac0_tx_clk_in emac0_gtx_clk emac0_tx_reset emac0_rx_reset emac0_md_clk — when **PHY Options** is set to "MDIO" i2cemac0_scl_in — when **PHY Options** is set to "I2C"
2	emac1 i2cemac1 — when **PHY Options** is set to "I2C"	emac_ptp_ref_clock emac1_rx_clk_in emac1_tx_clk_in emac1_gtx_clk emac1_tx_reset emac1_rx_reset emac1_md_clk — when **PHY Options** is set to "MDIO" i2cemac1_scl_in — when **PHY Options** is set to "I2C"
3	emac2 i2cemac2 — when **PHY Options** is set to "I2C"	emac_ptp_ref_clock emac2_rx_clk_in emac2_tx_clk_in emac2_gtx_clk emac2_tx_reset emac2_rx_reset emac2_md_clk — when **PHY Options** is set to "MDIO" i2cemac2_scl_in — when **PHY Options** is set to "I2C"

Table 29-44: SPIM Pins Interface Simulation Model

To FPGA	Conduit	Interface(s)
1	spim0	spim0_sclk_out
2	spim0 spim1	spim0_sclk_out spim1_sclk_out

Table 29-45: SPIS Pins Interface Simulation Model

To FPGA	Conduit	Interface(s)
1	spis0	spis0_sclk_out
2	spis0 spis1	spis0_sclk_out spis1_sclk_out

Table 29-46: UART Pins Interface Simulation Model

To FPGA	Conduit	Interface(s)
1	uart0	None
2	uart0 uart1	

Table 29-47: I²C Pins Interface Simulation Model

To FPGA	Conduit	Interface(s)
1	i2c0	i2c0_scl_in i2c0_clk
2	i2c0 i2c1	i2c0_scl_in i2c0_clk i2c1_scl_in i2c1_clk

Table 29-48: NAND Pins Interface Simulation Model

To FPGA	Conduit	Interface(s)
1	nand	None

Table 29-49: Trace Pins Interface Simulation Model

To FPGA	Conduit	Interface(s)
1	trace	trace_s2f_clk

Table 29-50: QSPI Pins Interface Simulation Model

To FPGA	Conduit	Interface(s)
1	qspi	None

Document Revision History

Table 29-51: Document Revision History

Date	Version	Changes
October 2016	2016.10.28	Maintenance release
May 2016	2016.05.27	Removed *FPGA EMAC Switch Interface* section. The application interface (also called the switch interface) is not supported.
May 2016	2016.05.03	Removed references to FPGA to HPS SDRAM simulation.
November 2015	2015.11.02	Added information about the signals on the "Advanced FPGA Placement" tab.
May 2015	2015.05.04	Maintenance release.
December 2014	2014.12.15	Maintenance release.
August 2014	2014.08.15	Initial release.

Booting and Configuration

A

2016.10.28

a10_5v4 Subscribe Send Feedback

The Altera system-on-a-chip (SoC) device provides a variety of ways to boot the hard processor system (HPS) and configure the FPGA portion.

Boot Overview

The boot flow of the HPS is a multi-stage process. Each stage is responsible for loading the next stage.

The first stage is the boot ROM execution. The boot ROM code, located in the HPS, brings the processor out of reset, puts the processor into a known and stable state, finds the second-stage boot loader and passes control to the next stage. The boot ROM code is only aware of the second-stage boot loader and not aware of any potential subsequent software stages. During this time, the boot ROM also seamlessly handles any error conditions that may occur.

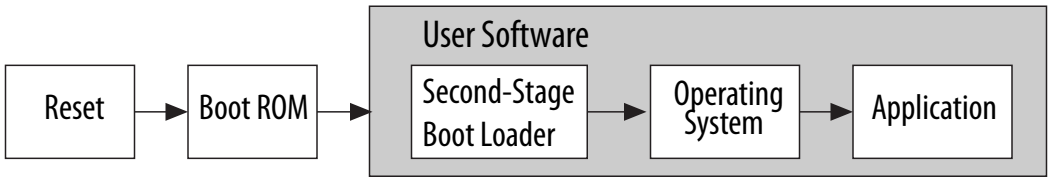
The next stage is when control passes to the second-stage boot loader. The second-stage boot loader is located external to the HPS, either in off-chip flash memory or within the FPGA. If the FPGA is used, the second stage boot loader can execute directly from the FPGA without the need to copy it to on-chip RAM. The second stage-boot loader locates and loads the next stage software and so on.

Before the control is passed to the second stage boot loader, it can be decrypted and/or authenticated if a secure boot is enabled.

After a warm reset, the user can program the HPS to instruct the boot ROM to find an image in the on-chip RAM and execute directly from that. In this case, the image that resides in RAM is unauthenticated and clear text, although it may have been imported into on-chip RAM as authenticated code initially.

The figure below illustrates the typical boot flow. However, there may be more or less software stages in the user software than shown and the roles of the software stages may vary.

Figure A-1: Typical Boot Flow



Related Information

SoC Security on page 6-1
For more information about secure boot, refer to the *SoC Security* chapter.

FPGA Configuration Overview

After power is applied to the FPGA and HPS portion of the SoC, configuration may begin. The FPGA may be configured through a variety of ways, such as through the HPS, JTAG, or an external host.

The Configuration Subsystem (CSS) in the FPGA portion of the device is responsible for obtaining an FPGA configuration image and configuring the FPGA fabric, as well as the IOCSRs (I/O Configuration and Status Registers). The I/O configuration file programs the following in the IOCSRs:

- FPGA I/O
- Shared I/O
- Hard memory controller I/O
- Hard memory controller settings, such as memory type, frequency and timings
- FPGA PLL settings
- Transceiver settings

The FPGA configuration ends when the configuration image has been fully loaded and the FPGA enters user mode. The FPGA configuration image is typically stored in external non-volatile flash-based memory. The FPGA Configuration Subsystem can obtain a configuration image from the HPS through the FPGA Manager or from any of the sources supported by the Arria 10 FPGA family.

Related Information

- **FPGA Manager** on page 4-1
  For more information regarding programming the FPGA through the HPS, refer to the *FPGA Manager* chapter.
- **Configuration, Design Security, and Remote System Upgrades in Arria 10 Devices**

Booting and Configuration Options

SoC initialization includes the booting of the HPS and the configuration of the FPGA fabric and I/O.

The I/O provided in the SoC are:

- HPS dedicated I/O that are configured by the HPS
- Shared I/O used by the HPS or FPGA and configured by the Quartus Prime I/O configuration file.
- FPGA I/O configured by the Quartus Prime I/O configuration file.
- Hard memory controller I/O shared by the HPS and FPGA and configured by the Quartus Prime I/O configuration file.

Depending on the initialization option you choose, the I/O configuration is handled differently. You can choose one of the three initialization options:

- The HPS boot and FPGA configuration occur separately.
- The HPS boots first and then configures the FPGA.
- The HPS boots from the FPGA after the FPGA is configured.

Note: The HPS and FPGA fabric must be powered at the same time. You must not power-down the HPS or FPGA independently in the middle of device operation.

The following three figures illustrate the possible HPS boot and FPGA configuration schemes. The arrows in the figures denote the data flow direction.

Figure A-2: Separate FPGA Configuration and HPS Booting

In the figure below, the FPGA configuration and HPS boot can occur separately. The FPGA obtains its configuration image from a non-HPS source. The source is sent to the CSS block which configures the FPGA fabric, FPGA I/O, shared I/O, hard memory controller I/O and other settings. If the hard memory controller or shared I/O are required by the HPS, then the FPGA fabric and I/O (FPGA, shared and hard memory controller) configuration must be complete before the HPS can boot.

The HPS boot ROM obtains its second-stage boot loader from a non-FPGA fabric source. The second-stage boot loader should contain the configuration for the HPS dedicated I/O.

For more information about FPGA configuration, refer to the "FPGA Configuration" section.

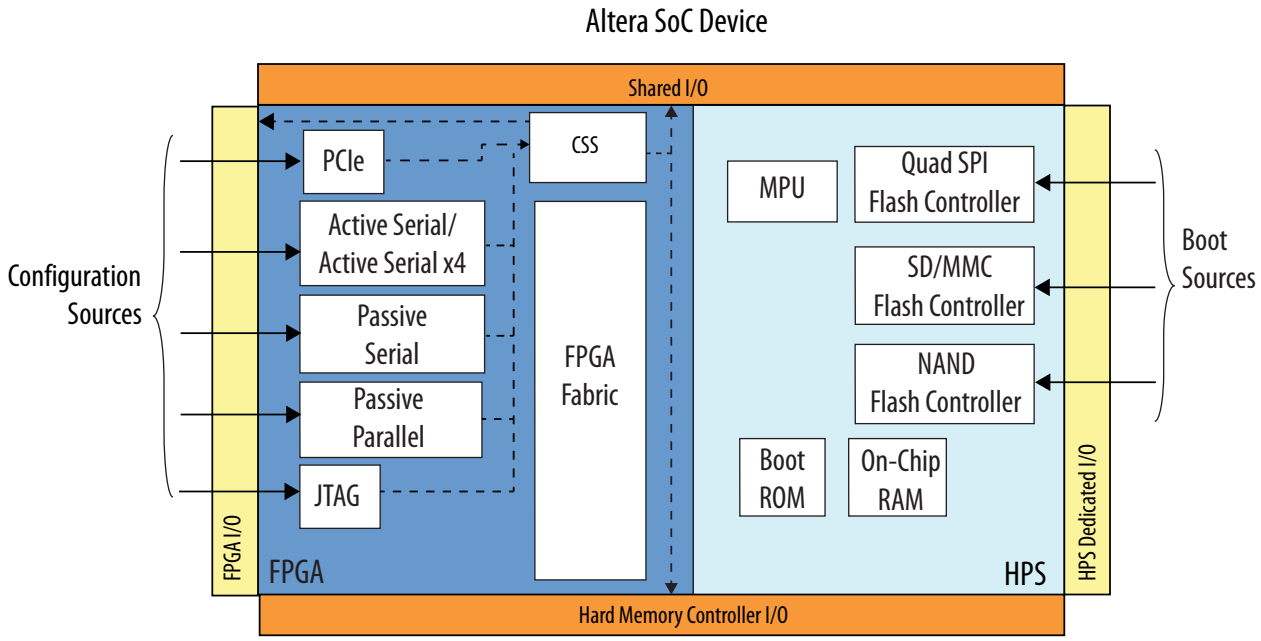


Figure A-3: HPS Boots First and then Configures the FPGA

In the figure below, the HPS boots first through one of its non-FPGA fabric boot sources. If the hard memory controller or shared I/O are required by the HPS during booting then you can either execute a full FPGA configuration flow or an early I/O release configuration flow. The FPGA must be in a power-on state for the HPS to reset properly and for the second stage boot loader to initiate configuration through the FPGA Manager. The software executing on the HPS obtains the FPGA configuration image from any of its flash memory devices. For more information about full FPGA configuration or early I/O release configuration after HPS booting, refer to the "FPGA Configuration" section.

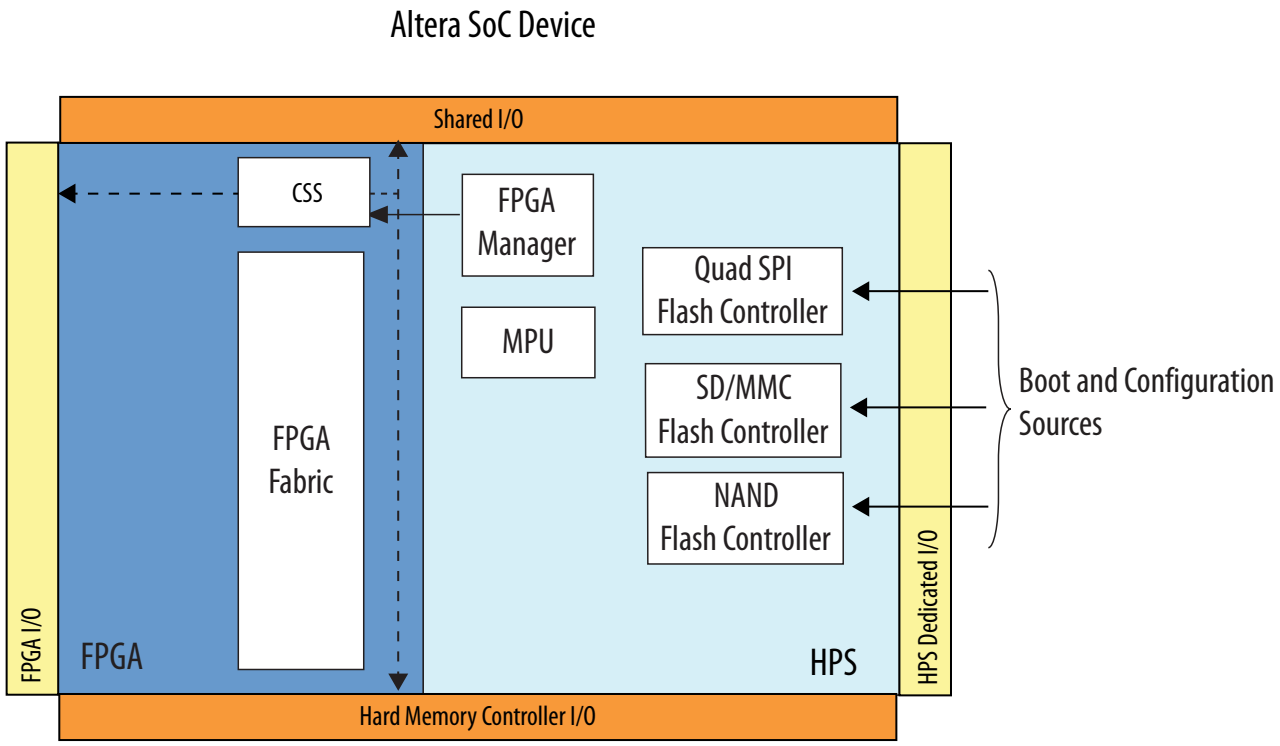
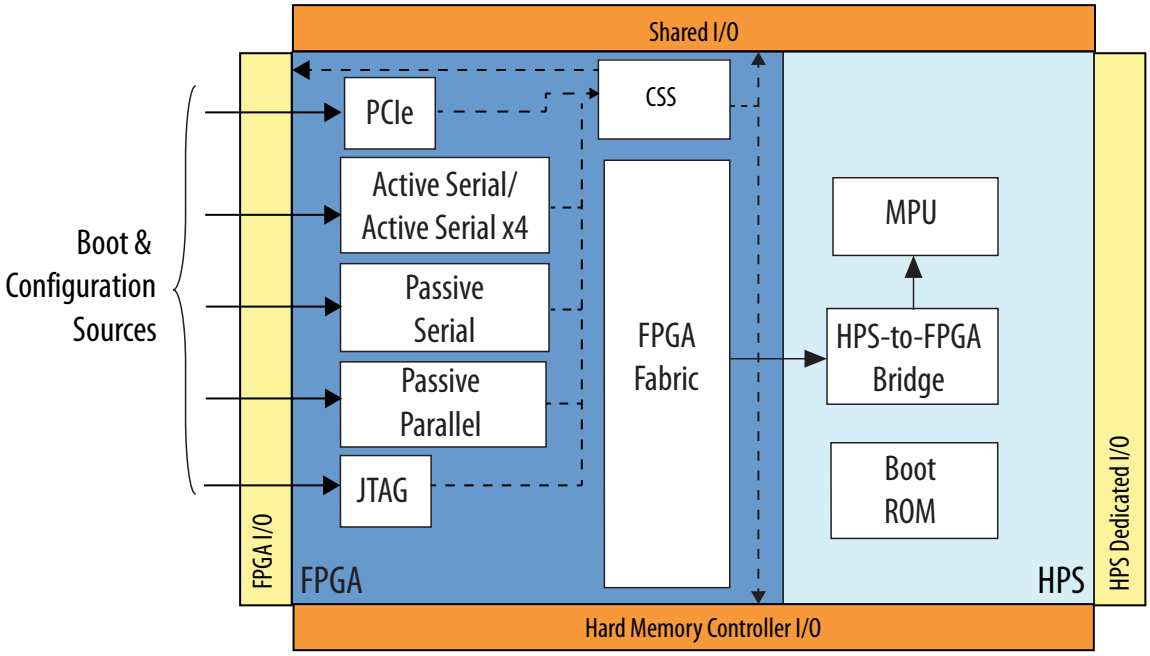


Figure A-4: HPS Boots From FPGA

In the figure below, the FPGA is configured first through one of its non-HPS configuration sources. The CSS block configures the FPGA fabric as well as the FPGA I/O, shared I/O and hard memory controller I/O. The HPS executes the second-stage boot loader from the FPGA. In this situation, the HPS should not be released from reset until the FPGA is powered on and programmed. Once the FPGA is in user mode and the HPS has been released from reset, the boot ROM code begins executing. The HPS boot ROM code executes the second-stage boot loader from the FPGA fabric over the HPS-to-FPGA bridge.

You can select a boot from FPGA by setting the BSEL value to 0x1. If the fpga_boot_f fuse is blown then the FPGA is always selected as the boot source. In both cases, when the FPGA is selected as the boot source, the CSEL fuses are ignored and clock configuration is controlled through the second-stage boot loader code in the FPGA.



Related Information

FPGA Configuration on page 30-46
For more information about full and early I/O release FPGA configuration

Boot Definitions

The following sections contain basic terms and definitions that are part of the boot process.

Reset

Reset precedes the boot stages and is an important part of device initialization. There are two different reset types: cold reset and warm reset.

The FPGA portion of the SoC can trigger a warm or cold reset on completion of configuration.

The boot process begins when CPU0 in the MPU exits from the reset state. When CPU0 exits from reset, it starts executing code at the reset exception address where the boot ROM code is located. CPU1 remains in reset during this time and is brought out of reset by user software.

With warm reset, some software registers are preserved and the boot process may skip some steps depending on software settings. In addition, on a warm reset, the second-stage boot loader has the ability to be executed from on-chip RAM.

Boot ROM

The boot ROM code is 128 KB in size and located in on-chip ROM at address range 0xFFFC0000 to 0xFFFDFFFF. The function of the boot ROM code is to determine the boot source, initialize the HPS after a reset, and jump to the second-stage boot loader. In the case of indirect execution, the boot ROM code loads the second-stage boot loader image from the flash memory to on-chip RAM. The boot ROM performs the following actions to initialize the HPS:

- Enable instruction cache, branch predictor, floating point unit, NEON vector unit of CPU0
- Sets up the level 4 (L4) watchdog 0 timer
- Initializes the flash controller to default settings
- Configures HPS dedicated I/O

When booting from flash memory, the boot ROM code uses the top 32 KB of the on-chip RAM as data workspace. This area is reserved for the boot ROM code after a reset until the boot ROM code passes software control to second-stage boot loader. The maximum second-stage boot loader size is 208 KB with authentication and 224 KB without. For a warm reset or cold reset with a boot from FPGA, the boot ROM code does not reserve the top 32 KB of the on-chip RAM, and the user may place user data in this area without being overwritten by the boot ROM.

The boot process begins when CPU0 exits from the reset state. The boot ROM code only executes on CPU0. CPU1 is held in reset while boot ROM executes on CPU0. When a CPU0 exits from reset, it starts executing code at the reset exception address.

During boot ROM execution, the clock control fuse information is automatically sent to the Clock Manager, the memory control fuse information is automatically sent to the Reset Manager and all other fuse functions (authentication, encryption, private and public key source, hash functions) are stored in a memory-mapped location for boot code to read. In normal operation, the boot ROM is mapped at the reset exception address so code starts executing in the boot ROM.

When CPU0 exits the boot ROM code and starts executing user software, the boot ROM access is disabled. The user software in CPU0 must map the user software exception vectors at 0x0 (which is previously mapped to boot ROM exception vectors). The user software also has the option of releasing CPU1 from reset. If CPU1 is released from reset, CPU1 executes the user software exception instead of boot ROM.

Boot Select

The boot select (BSEL) pins offer multiple methods to obtain the second-stage boot image. On a cold reset, the boot source is determined by a combination of secure boot fuses and BSEL pins. These fuse values and BSEL pin values are sent to the Security Manager module of the HPS when the cold reset occurs. When the HPS is released from reset, the boot ROM reads the `bootinfo` register of the System Manager to determine the source of the boot.

Note: If the fpga_boot_f fuse is blown, the BSEL pins are bypassed and the HPS can only boot from the FPGA. Additionally, the clock select (CSEL) fuse values are ignored and clock configuration is controlled through the FPGA. This configuration allows the HPS to boot from encrypted user code in the FPGA. If the boot source is the FPGA, the boot ROM code does not configure any of the

boot-specific HPS I/Os for booting from flash memory. If the fpga_boot_f fuse is not blown, then the boot source is determined according to the BSEL pins. If the BSEL pins are used for determining the boot source, then the following table shows the flash devices assigned to each encoding.

Note: If a boot from FPGA is required (BSEL[2:0]=0x1), then you must ensure that the HPS is not released from reset until after the FPGA has been fully programmed. Otherwise, the `bootinfo` register to determine the boot source might be read incorrectly by the boot ROM. FPGA readiness is indicated by handshake signals, `f2h_boot_from_fpga_ready` and `f2h_boot_from_fpga_on_failure`, from the FPGA to the HPS. The `f2h_boot_from_fpga_ready` signal must be pulled up to indicate readiness. Refer to the "Instantiating the HPS Component" chapter for more information about FPGA boot handshake signals.

Note: The acronyms BSEL and BOOTSEL are used interchangeably to define the boot select pins.

Table A-1: BSEL Values for Boot Source Selection

BSEL[2:0] Value	Flash Device
0x0	Reserved
0x1	FPGA (HPS-to-FPGA bridge)
0x2	1.8 V NAND flash memory
0x3	3.0 V NAND flash memory
0x4	1.8 V SD/MMC flash memory with external transceiver
0x5	3.0 V SD/MMC flash memory with internal transceiver
0x6	1.8 V quad SPI flash memory
0x7	3.0 V quad SPI flash memory

Note: If the BSEL value is set to 0x4 or 0x5, an external translation transceiver may be required to supply level-shifting and isolation if the SD cards interfacing to the SD/MMC controller must operate at a different voltage than the controller interface. Please refer to the *SD/MMC Controller* chapter for more information.

The typical boot flow is for the boot ROM code to find the second-stage boot loader image on a flash device, load that into on-chip RAM and execute it. After a warm reset, the boot ROM code can be instructed to find the image in RAM and execute that.

The HPS flash sources can store various file types, such as:

- FPGA programming files
- Second-stage boot loader binary file (up to four copies)
- Operating system binary files
- Application file system

The second-stage boot loader image in flash can be authenticated and decrypted by the HPS. A boot directly from the HPS on-chip RAM is always unauthenticated and in clear text, although it may have an optional CRC if required.

When the BSEL value is 0x1, the FPGA is selected as the boot source for that boot. This selection is not permanent as it is when the fpga_boot_f fuse is enabled. In both cases, the CSEL fuses are also ignored and the HPS must be held in reset until the FPGA is powered on and programmed to prevent the boot ROM from misinterpreting the boot source.

If an HPS flash interface has been selected to load the boot image, then the boot ROM enables and configures that interface before loading the boot image into on-chip RAM, verifying it and passing software control to the second-stage boot loader.

If the FPGA fabric is the boot source, the boot ROM code waits until the FPGA portion of the device is in user mode, and is ready to execute code and then passes software control to the second-stage boot loader in the FPGA RAM.

Related Information

- **SD/MMC Controller** on page 14-1
  Refer to the *SD/MMC Controller* chapter for more information regarding features and functionality.
- **SoC Security** on page 6-1
  For more information about secure boot, refer to the *SoC Security* chapter.
- **General Interfaces** on page 27-3
  For more information about FPGA boot handshake signals, refer to the "General Interfaces" section of the *Instantiating HPS Component* chapter.

Boot Source I/O Pins

The HPS has 17 dedicated I/O pins. Three are used as a clock, cold reset and warm reset pin. The warm reset signal is a bidirectional signal; a warm reset event can drive into the HPS, or a warm reset event can be generated by the HPS and drive out of this pin.

The remaining 14 dedicated I/O pins are used for boot devices as well as other peripherals. Three of these 14 dedicated I/O pins are sampled by software at either cold or warm reset and convey boot source information. The following table identifies the pin mux values for the boot source options.

Table A-2: Boot Source MUX Selects

This table identifies the dedicated HPS signals that are mapped to each boot interface and the mux select value each boot interface signal requires. Note that Clock Manager clock inputs are also assigned to MUX select 4 and are documented here for thoroughness.

Signal	MUX Select		
	14	8	4
HPS_DEDICATED_4	NAND_ADQ0	SDMMC_DATA0	QSPI_CLK
HPS_DEDICATED_5	NAND_ADQ1	SDMMC_CMD	QSPI_IO0
HPS_DEDICATED_6/BOOTSEL2	NAND_WE_N	SDMMC_CCLK	QSPI_SS0
HPS_DEDICATED_7	NAND_RE_N	SDMMC_DATA1	QSPI_IO1
HPS_DEDICATED_8	NAND_ADQ2	SDMMC_DATA2	QSPI_IO2_WPN
HPS_DEDICATED_9	NAND_ADQ3	SDMMC_DATA3	QSPI_IO3_HOLD
HPS_DEDICATED_10/BOOTSEL1	NAND_CLE	SDMMC_PWR_ENA	–
HPS_DEDICATED_11/BOOTSEL0	NAND_ALE	QSPI_SS1	CM_PLL_CLK0
HPS_DEDICATED_12	NAND_RB	SDMMC_DATA4	CM_PLL_CLK1
HPS_DEDICATED_13	NAND_CE_N	SDMMC_DATA5	CM_PLL_CLK2
HPS_DEDICATED_14	NAND_ADQ4	SDMMC_DATA6	CM_PLL_CLK3
HPS_DEDICATED_15	NAND_ADQ5	SDMMC_DATA7	CM_PLL_CLK4

Signal	MUX Select		
	14	8	4
HPS_DEDICATED_16	NAND_ADQ6	QSPI_SS2	–
HPS_DEDICATED_17	NAND_ADQ7	QSPI_SS3	–

Note: The MUX selects for the QSPI interface signals include both mux select 4 and mux select 8.

Table A-3: Boot Source MUX Selects (Alternate View)

This table displays the same information as the prior table but as an alternative view per interface selected. The number in each cell shows the mux select value needed to select the correct interface signal on the pin

Signal	Boot Interface		
	NAND	SDMMC	QSPI
HPS_DEDICATED_4	14	8	4
HPS_DEDICATED_5	14	8	4
HPS_DEDICATED_6/BOOTSEL2	14	8	4
HPS_DEDICATED_7	14	8	4
HPS_DEDICATED_8	14	8	4
HPS_DEDICATED_9	14	8	4
HPS_DEDICATED_10/BOOTSEL1	14	8	Not Used
HPS_DEDICATED_11/BOOTSEL0	14	Not Used	8
HPS_DEDICATED_12	14	8	Not Used
HPS_DEDICATED_13	14	8	Not Used
HPS_DEDICATED_14	14	8	Not Used
HPS_DEDICATED_15	14	8	Not Used
HPS_DEDICATED_16	14	Not Used	8
HPS_DEDICATED_17	14	Not Used	8

Related Information

- **Pin-Out Files for Arria 10 Devices**
- **Arria 10 Device Datasheet**
  For information on BSEL sampling after reset deassertion

Boot Fuses

During boot ROM execution, the boot ROM reads user-programmed fuses that configure the boot state of the HPS.

These fuse values are read by the Configuration Subsystem (CSS) after power-up and to the Security Manager where they are read by the boot ROM. The following list describes the basic fuses that affect the state of the HPS:

- Clock select fuses: Determine the clock frequencies of the MPU clock, the interconnect clocks and the peripheral clocks.
- Debug fuses: Determine the state of debug during the boot process and during hand-off to the second-stage boot loader.
- RAM fuses: Determine whether or not the various peripheral RAMs are cleared during a warm reset and how they are cleared (series or parallel).
- Boot fuses: Determines whether the second-stage boot loader is from on-chip RAM, whether an internal or external clock is used for booting and whether the boot source is solely from FPGA.
- Security fuses: Determines whether the second-stage boot loader is authenticated or decrypted. If authentication is used, then there are fuses that are read by the boot ROM to determine where the key authorization key (KAK) resides and what its length is.
- User fuses: The user fuses may either hold proprietary user information or may hold the value of the KAK.

Table A-4: `HPS_fusesec` Register Description

Bits	Name	Description
31:27	`Reserved`	Bit values in this field are undefined.
26:23	`csel_f`	This field indicates the value of the clock select fuses that are available for configuring the clock for the boot interface and for the PLLs. Refer to the *Clock Configuration* section for more information on CSEL encodings.
22	`dbg_access_f`	This fuse determines the initial state of the debug access domains.
21	`dbg_lock_JTAG`	This field indicates if the HPS JTAG access level can be changed through software when the HPS is released from reset. • 0x0= HPS JTAG access level can be changed through the `sec_jtagdbg` register. • 0x1= HPS JTAG access level cannot be changed (locked).
20	`dbg_lock_DAP`	This field indicates if the DAP access level can be changed through software when the HPS is released from reset. • 0x0= The DAP access level can be changed through the `sec_dapdbg` register. • 0x1= The DAP access level cannot be changed (locked).

Bits	Name	Description
19	dbg_lock_CPU0	This field indicates if the CPU0 debug access level can be changed through software when the HPS is released from reset. • 0x0= CPU0 debug access level can be changed through the sec_cpu0dbg register. • 0x1= CPU0 debug access level cannot be changed (locked).
18	dbg_lock_CPU1	This field indicates if the CPU1 debug access level can be changed through software when the HPS is released from reset. • 0x0= The CPU1 debug access level can be changed through the sec_cpu1dbg register. • 0x1= The CPU1 debug access level cannot be changed (locked).
17	dbg_lock_CS	This field indicates if the CoreSight debug access level can be changed through software when the HPS is released from reset. • 0x0= The CoreSight debug access level can be changed through the sec_csdbg register. • 0x1= The CoreSight debug access level cannot be changed (locked).
16	dbg_lock_FPGA	This field indicates if the FPGA debug access level can be changed through software when the HPS is released from reset. • 0x0= The FPGA debug access level can be changed through the sec_fpgadbg register. • 0x1= The FPGA debug access level cannot be changed (locked).
15:12	Reserved	Bit values in this field are undefined.
11	clr_ram_order_f	This fuse value determines how RAMs are cleared during a tamper event. • 0x0= All RAMs are cleared in parallel. • 0x1= All RAMs are cleared in series.

Bits	Name	Description
10	clr_ram_cold_f	This fuse value indicates what happens to the RAM on a cold reset. • 0x0= All RAMs are not cleared on a cold reset. • 0x1= All RAMs are cleared on a cold reset.
9	clr_ram_warm_f	This fuse value indicates what happens to the RAMs on a warm reset. • 0x0= All RAMs are not cleared on a warm reset. • 0x1= All RAMs are cleared on a warm reset.
8	oc_boot_f	This fuse value determines if the second-stage boot code is allowed to boot from on-chip RAM. • 0x0= Second-stage boot can be from on-chip RAM if enabled by the System Manager. • 0x1= Second-stage boot is not from on-chip RAM.
7	hps_clk_f	This fuse value selects the clock used for the boot process and in the case of a tamper event, memory scrambling. • 0x0= The external oscillator, EOSC1, is used for boot. • 0x1= The internal oscillator, cb_intosc_ls_clk, is used for boot.
6	fpga_boot_f	If blown, this fuse value allows the FPGA to configure independently and allows the HPS to boot from an encrypted next-stage boot source that was decrypted into the FPGA. • 0x0= Booting is dependent on the BSEL pins. • 0x1= HPS only boots from the FPGA; BSEL options are ignored and CSEL fuse options are ignored.

Bits	Name	Description
5	aes_en_f	This fuse value indicates if a decryption of the flash image is always performed. • 0x0= An AES decryption of the flash image is determined from the second stage boot loader header. • 0x1= An AES decryption of the flash image is always performed.
4:2	kak_src_f	This bit field indicates the source of the Key Authorization Key (KAK) which can be in: • Proprietary ROM • FPGA logic elements • User fuses
1	kak_len_f	This fuse value indicates the Key Authorization Key (KAK) length: • 0x0= 256 bits • 0x1= 384 bits
0	authen_en_f	This fuse value determines whether authentication of flash images is required prior to execution. • 0x0= No authentication of the flash image is required prior to execution. • 0x1= HPS authentication of all flash images is required prior to execution.

For more information about the fuses and how they work, refer to the *SoC Security* chapter.

Related Information

SoC Security on page 6-1
For more information about secure boot, refer to the *SoC Security* chapter.

Flash Memory Devices for Booting

The memory controllers and devices that contain the boot loader image have configuration requirements for proper boot from flash.

On all flash devices, there is an area of memory, called the boot area that contains up to four second-stage boot loader images. For the QSPI and SD/MMC devices, the boot area is 1 MB in size. For NAND devices the boot area is four device blocks in size and may be larger than 1 MB if the NAND erase block is larger than 256 KB.

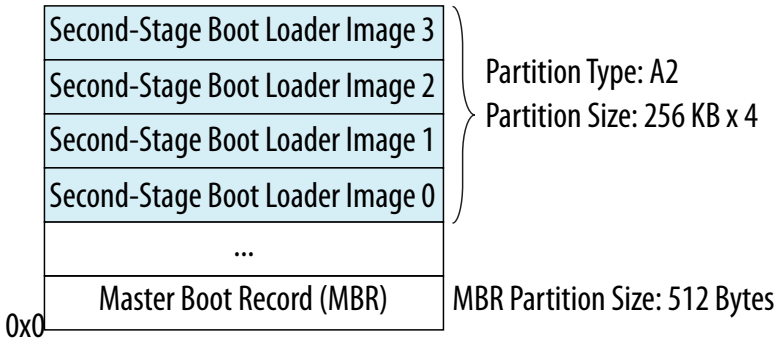
The SD/MMC, Quad SPI and NAND flash devices all support raw and MBR (partition) mode. In raw mode, the boot image is located at the start of the flash memory device, at offset 0x0. In MBR mode:

- The boot image is read from a custom partition (0xA2)
- The first image is located at the beginning of the partition, at offset 0x0
- Start address = partition start address

SD/MMC Flash Devices

The following figure shows an SD/MMC flash image layout example for boot. The master boot record (MBR) is located in the first 512 bytes of the memory. The MBR contains information about the partitions (address and size of partition). The second-stage boot loader image is stored in partition A2. Partition A2 is a custom raw partition with no file system.

Figure A-5: SD/MMC Flash Image Layout



The SD/MMC controller supports two booting modes:

- MBR (partition) mode
  - The boot image is read from a custom partition (0xA2)
  - The first image is located at the beginning of the partition, at offset 0x0
  - Start address = partition start address
- Raw mode
  - If the MBR signature is not found, SD/MMC driver assumes it is in raw mode
  - The boot image data is read directly from sectors in the user area and is located at the first sector of the SD/MMC
  - The first image is located at the start of the memory card, at offset 0x0
  - Start address = 0x0

The MBR contains the partition table, which is always located in the first sector (LBA0) with a memory size of 512 bytes. The MBR consists of executable code, four partition entries, and the MBR signature. A MBR can be created by specific tools like the FDISK program.

Table A-5: MBR Structure

Offset	Size (In Bytes)	Description
0x000	446	Code area
0x1BE	16	Partition entry for partition 1
0x1CE	16	Partition entry for partition 2
0x1DE	16	Partition entry for partition 3
0x1EE	16	Partition entry for partition 4
0x1FE	2	MBR signature: 0xAA55

The standard MBR structure contains a partition with four 16-byte entries. Thus, memory cards using this standard table cannot have more than four primary partitions or up to three primary partitions and one extended partition.

Each partition type is defined by the partition entry. The boot images are stored in a primary partition with custom partition type (0xA2). The SD/MMC flash driver does not support a file system, so the boot images are located in partition A2 at fixed locations.

Table A-6: Partition Entry

Offset	Size (In Bytes)	Description
0x0	1	Boot indicator. 0x80 indicates that it is bootable.
0x1	3	Starting CHS value
0x4	1	Partition type
0x5	3	Ending CHS value
0x8	4	LBA of first sector in partition
0xB	4	Number of sectors in partition

The boot ROM code configures the SD/MMC controller to default settings for the supported SD/MMC flash memory.

Related Information

SD/MMC Controller on page 14-1
Refer to the *SD/MMC Controller* chapter for more information regarding features and functionality.

Default Settings of the SD/MMC Controller

Table A-7: SD/MMC Controller Default Settings

Parameter		Default	Register Value
Card type		1 bit	The card type register (`ctype`) in the SD/MMC controller registers (`sdmmc`) = 0x0
Bus mode		—	SD/MMC[63]
Timeout		Maximum	The timeout register (`tmout`) = 0xFFFFFFFF
FIFO threshold RX watermark level		1	The RX watermark level field (`rx_wmark`) of the FIFO threshold watermark register (`fifoth`) = 0x1
Clock source		0	The clock source register (`clksrc`) = 0x0
Block size		512	The block size register (`blksiz`) = 0x200
Clock divider	Identification mode	32	The clock divider register (`clkdiv`)= 0x10 (2*16=32)
	Data transfer mode	Bypass	The clock divider register (`clkdiv`)= 0x00

(63) SPI cards are not supported for boot.

Parameter	Default	Register Value
External Device Power enable	Disabled (Power Off)	The `power_enable` bit in the `pwren` register is programmed to 0x0 out of reset. In Arria 10, the SD/MMC power enable is inverted. To compensate for this active low polarity, you can implement one of three options: • Force the power enable high on the board. • Use a GPIO to control the power enable. • Invert the power enable line on the board so that when software disables the power (`SDMMC_PWR_ENA_HPS` is high), the board inverts the signal to turn off the card.

CSEL Settings for the SD/MMC Controller

Table A-8: SD/MMC Controller Clock Options Based on CSEL and HPS_CLK fuse settings

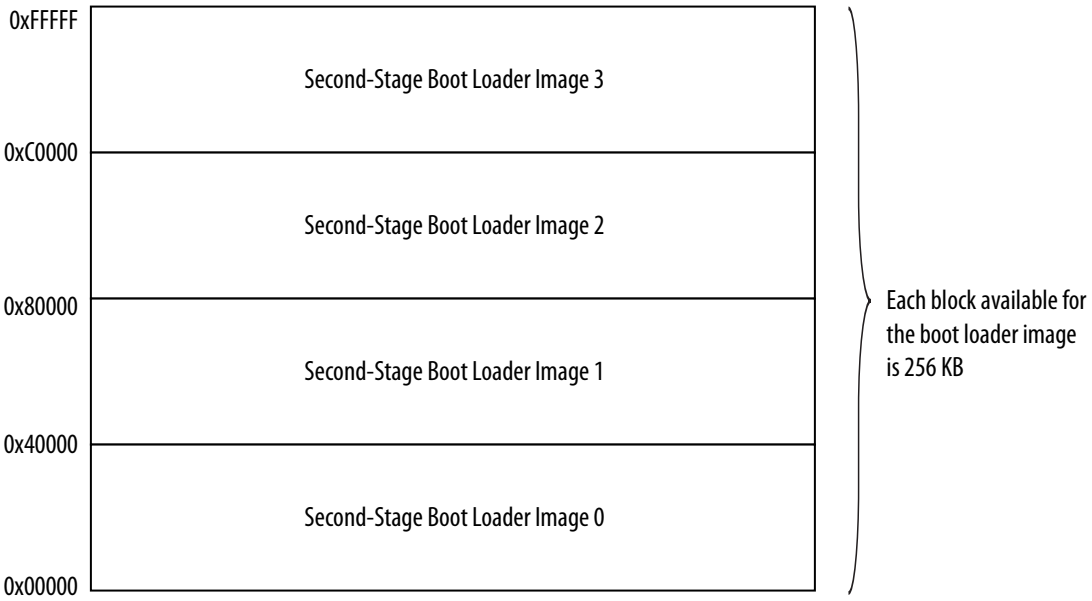
CSEL [3:0] Fuse Values	HPS CLK Fuse Value	Required Input Clock Range	Controller Clock	Controller Clock Frequency	ID Mode: Baud Rate Divisor	ID Mode: Device Clock	Data Transfer Mode: Baud Rate Divisor	Data Transfer Mode: Device Clock	PLL Status
0x0-0x1 and 0x3-0xF	1	60-200 MHz (Secure Bypass)	`cb_intosc_hs_clk/4`	15-50 MHz	128	29-97.75 KHz	4	937.5 KHz-3.125 MHz	Bypass
0x2	1	30-100 MHz (Secure PLL)	`cb_intosc_ls_clk/8`	2.75-12.5 MHz	32	29-97.75 KHz	1 (Bypass)	687.5 KHz-3.125 MHz	Locked
0x1	0	10-50 MHz (Untouched PLL)	`osc1_clk/4` (`EOSC1` pin)	2.5-12.5 MHz	32	19.5-97.75 KHz	1 (Bypass)	625 KHz-3.125 MHz	Untouched

CSEL [3:0] Fuse Values	HPS CLK Fuse Value	Required Input Clock Range	Controller Clock	Controller Clock Frequency	ID Mode: Baud Rate Divisor	ID Mode: Device Clock	Data Transfer Mode: Baud Rate Divisor	Data Transfer Mode: Device Clock	PLL Status
0x0 and 0x2-0x6	0	10-50 MHz (PLL Bypass)	osc1_clk/4 (EOSC1 pin)	2.5-12.5 MHz	32	19.5-97.75 KHz	1 (Bypass)	625 KHz-3.125 MHz	Bypass
0x7	0	10-15 MHz	osc1_clk*0.8333	8.333-12.5 MHz	32	65-97.75 KHz	1 (Bypass)	2.08-3.125 MHz	Locked
0x8	0	15-20 MHz	osc1_clk*0.625	9.375-12.5 MHz	32	73-97.75 KHz	1 (Bypass)	2.34-3.125 MHz	Locked
0x9	0	20-25 MHz	osc1_clk/2	10-12.5 MHz	32	78.13-97.75 KHz	1 (Bypass)	2.5-3.125 MHz	Locked
0xA	0	25-30 MHz	osc1_clk*0.4166	10.4175-12.5 MHz	32	81.25-97.75 KHz	1 (Bypass)	2.60-3.125 MHz	Locked
0xB	0	30-35 MHz	osc1_clk*0.3571	10.715-12.5 MHz	32	83.5-97.75 KHz	1 (Bypass)	2.68-3.125 MHz	Locked
0xC	0	35-40 MHz	osc1_clk*0.3125	10.9375-12.5 MHz	32	85.25-97.75 KHz	1 (Bypass)	2.75-3.125 MHz	Locked
0xD	0	40-45 MHz	osc1_clk*0.2777	11.111-12.5 MHz	32	86.75-97.75 KHz	1 (Bypass)	2.78-3.125 MHz	Locked
0xE	0	45-50 MHz	osc1_clk/4	11.25-12.5 MHz	32	87.75-97.75 KHz	1 (Bypass)	2.81-3.125 MHz	Locked

NAND Flash Devices

The NAND subsystem reserves at least the first 1 MB on the NAND device. If the NAND flash device has blocks greater than 256 KB, then the NAND subsystem reserves the first four blocks on the device. For a NAND device with less than 256 KB block size, the second-stage boot loader image must be placed in multiple blocks. The NAND subsystem expects to find up to four second-stage boot loader images on the NAND device. You may have less than four images if required. The second-stage boot loader image should always be at the start of a physical page. Because a block is the smallest area used for erase operation, any update to a particular image does not affect other images.

Figure A-6: NAND Flash Image Layout for 256 KB Memory Blocks



Related Information

NAND Flash Controller on page 13-1
Refer to the *NAND Flash Controller* chapter for more information regarding features and functionality.

NAND Flash Driver Features Supported in the Boot ROM Code

Table A-9: NAND Flash Support Features

Feature	Driver Support
Device	Open NAND Flash Interface (ONFI) 1.0 raw NAND or electronic signature devices, single layer cell (SLC) and multiple layer cell (MLC)
Chip select	CS0 only. Only CS0 is available to the HPS, the other three chip selects are routed out to the FPGA portion of the device
Bus width	8-bit only
Page size	2 KB, 4 KB, or 8 KB
Page per block	32, 64, 128
ECC	512-bytes with 8-bit correction

CSEL Settings for the NAND Controller

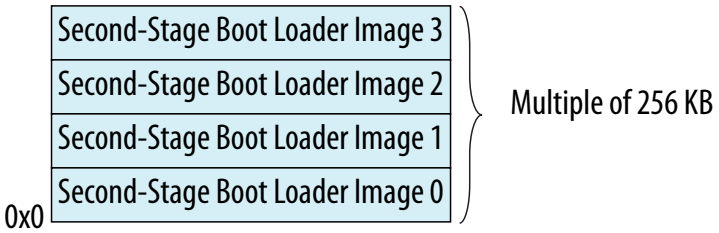
Table A-10: NAND Controller Clock Options Based on CSEL and HPS_CLK fuse settings

CSEL [3:0] Fuse Values	HPS CLK Fuse Value	Required Input Clock Range	l4_mp_clk	l4_mp_clk Frequency	nand_clk	PLL Status
0x0-0x1 and 0x3-0xF	1	60-200 MHz (Secure bypass)	cb_intosc_hs_clk	60-200 MHz	15-50 MHz	Bypassed
0x2	1	30-100 MHz (Secure PLL)	cb_intosc_ls_clk*2	60-200 MHz	15-50 MHz	Locked
0x1	0	10-50 MHz (Untouched PLL)	osc1_clk (EOSC1 pin)	10-50 MHz	2.50-12.5 MHz	Untouched
0x0 and 0x2-0x6	0	10-50 MHz (PLL bypass)	osc1_clk (EOSC1 pin)	10-50 MHz	2.50-12.5 MHz	Bypassed
0x7	0	10-15 MHz	osc1_clk*13.33	133.33-200 MHz	33.33-50 MHz	Locked
0x8	0	15-20 MHz	osc1_clk*10	150-200 MHz	37.50-50 MHz	Locked
0x9	0	20-25 MHz	osc1_clk*8	160-200 MHz	40-50 MHz	Locked
0xA	0	25-30 MHz	osc1_clk*6.66	166.66-200 MHz	41.66-50 MHz	Locked
0xB	0	30-35 MHz	osc1_clk*5.71	171.43-200 MHz	42.85-50 MHz	Locked
0xC	0	35-40MHz	osc1_clk*5	175-200 MHz	43.75-50 MHz	Locked
0xD	0	40-45MHz	osc1_clk*4.44	177.78-200 MHz	44.45-50 MHz	Locked
0xE	0	45-50MHz	osc1_clk*4	180-200 MHz	45-50 MHz	Locked

Quad SPI Flash Devices

The figure below shows the quad SPI flash image layout. The second-stage boot loader image is always located at offsets that are multiples of 256 KB.

Figure A-7: Quad SPI Flash Image Layout



The boot ROM code configures the quad SPI controller to default settings for the supported SPI or quad SPI flash memory.

Related Information

Quad SPI Flash Controller on page 15-1
Refer to the *Quad SPI Flash Controller* chapter for more information regarding features and functionality.

Quad SPI Controller Default Settings

Table A-11: Quad SPI Controller Default Settings

Parameter	Default Setting	Register Value
SPI baud rate	Divide by 4	The master mode baud rate divisor field (`bauddiv`) of the quad SPI configuration register (`cfg`) in the quad SPI controller registers (`qspiregs`) = 1.
Read opcode	Refer to the "QSPI Controller Clock Options Based on CSEL and HPS_CLK Fuse Settings" table to determine the correct configuration of the HPS_CLK and CSEL fuses for Normal or Fast Reads.	The read opcode in non-XIP mode field (`rdopcode`) in the device read instruction register (`devrd`) = 0x3 (for normal read) and 0xB (for fast read).
Instruction type	Single I/O (1 bit wide)	The address transfer width field (`addrwidth`) and data transfer width field (`datawidth`) of the `devrd` register = 0.

Parameter	Default Setting	Register Value
Number of address bytes	3 bytes	The number of address bytes field (`numaddrbytes`) of the device size register (`devsz`) = 2. **Note:** Before a reset, you must ensure that the QSPI flash device is configured to 3 byte address mode for the boot ROM to function properly.
Delay in terms of `l4_main_clk` for the length that the master mode chip select outputs are deasserted between words when the clock phase is zero	The default setting in clock cycles must equal 200 ns	The clock delay for chip select deassert (field `nss` in the quad SPI device delay register (`delay`)). Refer to `delay[31:24]` in the "Quad SPI Flash Delay Configuration" table in the "Quad SPI Flash Delay Configuration" section for calculations.
Delay in terms of `l4_main_clk` clock cycles between one chip select being deactivated and the activation of another. This delay ensures a quiet period between the selection of two different slaves and requires the transmit FIFO to be empty	The default setting is 0 clock cycles.	The clock delay for chip select deactivation (field `btwn` in the `delay` register = 0x0).
Delay in terms of `l4_main_clk` clock cycles between the last bit of the current transaction and the first bit of the next transaction. If the clock phase is zero, the first bit of the next transaction refers to the cycle in which the chip select is deselected	The default setting in clock cycles must equal 20 ns.	The clock delay for last transaction bit (field `after` in the `delay` register). Refer to `delay[15:8]` in the "Quad SPI Flash Delay Configuration" table in the "Quad SPI Flash Delay Configuration" section for calculations.
Added delay in terms of `l4_main_clk` clock cycles between setting `qspi_n_ss_out` low and first bit transfer	The default setting in clock cycles must equal 20 ns.	The clock delay of `qspi_n_ss_out` (field `init` in the `delay` register). Refer to `delay[7:0]` in the "Quad SPI Flash Delay Configuration" table in the "Quad SPI Flash Delay Configuration" section for calculations.

Quad SPI Flash Delay Configuration

The `delay` register in the quad SPI controller configures relative delay of the generation of the master output signals. All timings are defined in cycles of `l4_main_clk`.

The quad SPI flash memory must meet the following timing requirements:

- T_{SLCH}: 20 ns
  - T_{SLCH} is used to calculate the `init` field (`delay[7:0]`) in the `delay` register. The `init` field represents the delay in `l4_main_clk` clocks between pulling the device chip select (`qspi_n_ss_out`) low and the first bit transfer.
- T_{CHSH}: 20 ns
  - T_{CHSH} is used to calculate the `after` field (`delay[15:8]`) in the `delay` register. The `after` field represents the delay in the `l4_main_clk` clocks between last bit of the current transaction and the deassertion of the device chip select (`qspi_n_ss_out`).
- T_{SHSL}: 200 ns
  - T_{SHSL} is used to calculate the `nss` (`delay[31:24]`) field in the `delay` register and is the delay in the `l4_main_clk` clocks for the length that the master mode chip select outputs are deasserted between transactions.
- $T_{qspi\_ref\_clk}$ is the master reference clock/external clock, `l4_main_clk`.

The formulas to calculate the fields in the `delay` register are:

`delay[7:0]`= `init` = $T_{SLCH}/T_{qspi\_ref\_clk}$

`delay[15:8]`= `after` = $T_{CHSH}/T_{qspi\_ref\_clk}$

`delay[31:24]`= `nss` = $(T_{SHSL} - T_{qspi\_clk})/T_{qspi\_ref\_clk}$

Table A-12: Quad SPI Flash Delay Configuration for Device Delay (`delay`) Register

CSEL [3:0] Fuse Values	HPS CLK Fuse Value	Input Clock Range	$T_{qspi\_ref\_clk}$ (ns)	T_{qspi_clk} (ns)	`delay[7:0]` (init)	`delay[15:8]` (after)	`delay[31:24]` (nss)
0x0-0x1 and 0x3-0xF	1	60-200 MHz (Secure Bypass)	5	20	4	4	36
0x2	1	30-100 MHz (Secure PLL)	2.5	10	8	8	76
0x1	0	10-50 MHz (Untouched PLL)	20	80	1	1	6
0x0 and 0x2-0x6, 0xF	0	10-50 MHz (PLL Bypass)	20	80	1	1	6
0x7	0	10-15 MHz	2.5	10	8	8	76
0x8	0	15-20 MHz	2.5	10	8	8	76
0x9	0	20-25 MHz	2.5	10	8	8	76
0xA	0	25-30 MHz	2.5	10	8	8	76
0xB	0	30-35 MHz	2.5	10	8	8	76

CSEL [3:0] Fuse Values	HPS CLK Fuse Value	Input Clock Range	$T_{qspi\_ref\_clk}$ (ns)	T_{qspi_clk} (ns)	delay[7:0] (init)	delay[15:8] (after)	delay[31:24] (nss)
0xC	0	35-40 MHz	2.5	10	8	8	76
0xD	0	40-45 MHz	2.5	10	8	8	76
0xE	0	45-50 MHz	2.5	10	8	8	76

Read data capture delay is also configured when booting from QSPI. Depending on the CSEL and HPS_CLK fuse values, the Boot ROM configures the `delay` field of the `rddatacap` register in the QSPI differently. When you configure the CSEL and HPS_CLK fuses to enable the PLL, the boot ROM calibrates the interface by reading the QSPI signal for all delay values of the `rddatacap` register. The Boot ROM analyzes all of the delay values that return a valid signature and uses the delay value in the middle of the valid window as the value it programs into the `delay` field of the `rddatacap` register. If the PLL is not enabled, then the Boot ROM leaves the `rddatacap` register untouched.

Quad SPI Controller CSEL Settings

Table A-13: QSPI Controller Clock Options Based on CSEL and HPS_CLK Fuse Settings

CSEL [3:0] Fuse Values	HPS CLK Fuse Value	Required Input Clock Range	Controller clock	Controller Clock Frequency (l4_main_clock)	Baud Rate Divisor	External Device Clock (Divided down reference clock)	Default Flash Read Instruction	PLL Status
0x0-0x1 and 0x3-0xF	1	60-200 MHz (Secure Bypass)	cb_intosc_hs_clk	60-200 MHz	16	3.75-12 MHz	READ	Bypassed
0x2	1	30-100 MHz (Secure PLL)	cb_intosc_ls_clk*4	120-400 MHz	8	15-50 MHz	FAST_READ	Locked
0x1	0	10-50 MHz (Untouched PLL)	osc1_clk (EOSC1 pin)	10-50 MHz	4	2.50-12.5 MHz	READ	Untouched
0x0-0x6, 0xF	0	10-50 MHz (PLL Bypass)	osc1_clk (EOSC1 pin)	10-50 MHz	4	2.50-12.5 MHz	READ	Bypassed
0x7	0	10-15 MHz	osc1_clk*26.75	266.67-400 MHz	8	33.33-50 MHz	FAST_READ	Locked
0x8	0	15-20 MHz	osc1_clk*20	300-400 MHz	8	37.50-50 MHz	FAST_READ	Locked

CSEL [3:0] Fuse Values	HPS CLK Fuse Value	Required Input Clock Range	Controller clock	Controller Clock Frequency (l4_main_clock)	Baud Rate Divisor	External Device Clock (Divided down reference clock)	Default Flash Read Instruction	PLL Status
0x9	0	20-25 MHz	osc1_clk*16	320-400 MHz	8	40-50 MHz	FAST_READ	Locked
0xA	0	25-30 MHz	osc1_clk*13.33	333.33-400 MHz	8	41.66-50 MHz	FAST_READ	Locked
0xB	0	30-35 MHz	osc1_clk*11.42	42.85-50 MHz	8	42.85-50 MHz	FAST_READ	Locked
0xC	0	35-40 MHz	osc1_clk*10	350-400 MHz	8	43.75-50 MHz	FAST_READ	Locked
0xD	0	40-45 MHz	osc1_clk*8.88	355.55-400 MHz	8	44.45-50 MHz	FAST_READ	Locked
0xE	0	45-50 MHz	osc1_clk*3	360-400 MHz	8	45-50 MHz	FAST_READ	Locked

Clock Select

The boot ROM reads the clock select values to determine what frequency has been selected for the CPU clock and any interface clock during boot.

If the FPGA boot fuse is not blown, the clock select (CSEL) fuses are used to configure the main PLL and peripheral PLL. The Clock Manager samples the clock configuration on a cold and warm reset. If the hps_clk_f fuse is blown, the internal oscillator divided by 2, `cb_intosc_hs_div2_clk`, is used as the boot clock. If the fuse is not blown, an external oscillator is used. During boot ROM execution, the boot code configures the device clock based on the CSEL fuse settings or software code. The `cb_intosc_hs_div2_clk` can be considered the secure reference clock option.

Note: The terms CSEL and CLKSEL are used interchangeably in Altera documentation to refer to clock select.

The CSEL settings are not used to configure the PLLs for the following conditions:

- A boot from RAM (warm reset)
- A boot from FPGA
- The clock select is set to 0x1, indicating the clocks should not be touched.

If a bypass condition is encountered, the boot ROM checks to make sure that the boot clock has a proper source according to the security level set and if it does not, the boot ROM stalls. ROM code bypasses the PLLs at warm or cold reset under the following conditions:

- The hps_clk_f fuse is not blown and the CSEL fuse value is 0xF. This setting is an unsecure bypass that uses the external oscillator (EOSC1) as the clock source.
- The hps_clk_f fuse is not blown and the CSEL fuses values are not 0xF, 0x1 or one of the values from 0x7 to 0xE that indicates PLL use. These are mapped to unsecure bypass using the external oscillator (`EOSC1`) as the clock source.
- The hps_clk fuse is set to secure clock, but the CSEL fuse value is not 0x2. This setting is considered secure bypass, using the internal oscillator as the clock source.

CSEL Fuse Encodings During Boot

The intent of the CSEL fuses is to give the user more control over the CPU clock in both non-secure and secure mode. For example, if you select CSEL=0x0 (the default) in non-secure mode, then the boot ROM configures all the clocks back to their default bypass (boot) mode and the MPU is clocked by the `EOSC1` (10-50 MHz). However, if you have a 10 MHz external oscillator (`EOSC1`) and want to operate faster, then the CSEL fuses can be programmed to 0x7 for a MPU clock of 553 MHz. Similarly, if the input clock is 25 MHz and the CSEL fuses are programmed to 0x9 then the MPU clock is 800 MHz (25 * 32). If CSEL=0xA is chosen, then the same clock gives a MPU clock of 650 MHz (25 * 26). Finally, if some set of CSEL fuses have been blown, you can return to the default bypass state (CSEL=0x0) by blowing all the fuses (CSEL=0xF).

The following tables represent the clock frequencies that are available during the boot process according the boot clock that is selected. As noted previously, when the internal secure clock, `cb_intosc_ls_clk`, is used as the boot clock, the meaning of the CSEL fuses is slightly different.

To configure the clock speeds for second-stage boot loader execution, you need to select the clock speeds in the HPS component of Qsys. The tables below only apply to the boot process.

Table A-14: CSEL Encodings for hps_clk_f = 0 (Non-secure Operation)

CSEL[3:0] Fuse Value	Description	MPU Clock Value
0x0	When no fuses are blown, the boot ROM returns clocks back to their default (bypass) boot mode and the CPU is driven by the external oscillator (`EOSC1`), which must be in the range of 10 to 50 MHz. No PLL is enabled.	External Oscillator, `EOSC1` (10 to 50 MHz)
0x1	All the clock codes are bypassed and the clock source is user selected.	User-selected clock source
0x2	Reserved	External Oscillator, `EOSC1` (10 to 50 MHz)
0x3	Reserved	External Oscillator, `EOSC1` (10 to 50 MHz)
0x4	Reserved	External Oscillator, `EOSC1` (10 to 50 MHz)
0x5	Reserved	External Oscillator, `EOSC1` (10 to 50 MHz)
0x6	Reserved	External Oscillator, `EOSC1` (10 to 50 MHz)

CSEL[3:0] Fuse Value	Description	MPU Clock Value
0x7	External oscillator input clock, EOSC1, is in the range of 10 to 15 MHz	533 to 800 MHz
0x8	External oscillator input clock, EOSC1, is in the range 15 to 20 MHz.	600 to 800 Mhz
0x9	External oscillator input clock, EOSC1, is in the range 20 to 25 MHz.	640 to 800 Mhz
0xA	External oscillator input clock, EOSC1, is in the range 25 to 30 MHz.	666 to 800 Mhz
0xB	External oscillator input clock, EOSC1, is in the range 30 to 35 MHz.	685 to 800 Mhz
0xC	External oscillator input clock, EOSC1, is in the range 35 to 40 MHz.	700 to 800 Mhz
0xD	External oscillator input clock, EOSC1, is in the range 40 to 45 MHz.	711 to 800 Mhz
0xE	External oscillator input clock, EOSC1, is in the range 45 to 50 MHz.	720 to 800 Mhz
0xF	The boot ROM returns clocks back to their default (bypass) boot mode and the CPU is driven by the external oscillator (EOSC1). No PLL is enabled.	External Oscillator (10 to 50 MHz)

Table A-15: CSEL Encodings for hps_clk_f = 1 (Secure Operation)

CSEL[3:0] Fuse Value	Description	MPU Clock Value
0x0	Bypass mode; in this mode all clocks are reset and boot mode is ensured active; cb_intosc_hs_div2_clk (cb_intosc_hs clock divided by 2) is used.	cb_intosc_hs_div2_clk (60 to 200 MHz)
0x1	All the clock codes are bypassed and the clock is user selected.	User-selected clock source
0x2	PLL is used with the internal oscillator (30 to 100 MHz)	120 to 800 MHz

CSEL[3:0] Fuse Value	Description	MPU Clock Value
0x3	Reserved	cb_intosc_hs_div2_clk (60 to 200 MHz)
0x4	Reserved	cb_intosc_hs_div2_clk (60 to 200 MHz)
0x5	Reserved	cb_intosc_hs_div2_clk (60 to 200 MHz)
0x6	Reserved	cb_intosc_hs_div2_clk (60 to 200 MHz)
0x7	Reserved	cb_intosc_hs_div2_clk (60 to 200 MHz)
0x8	Reserved	cb_intosc_hs_div2_clk (60 to 200 MHz)
0x9	Reserved	cb_intosc_hs_div2_clk (60 to 200 MHz)
0xA	Reserved	cb_intosc_hs_div2_clk (60 to 200 MHz)
0xB	Reserved	cb_intosc_hs_div2_clk (60 to 200 MHz)
0xC	Reserved	cb_intosc_hs_div2_clk (60 to 200 MHz)
0xD	Reserved	cb_intosc_hs_div2_clk (60 to 200 MHz)
0xE	Reserved	cb_intosc_hs_div2_clk (60 to 200 MHz)
0xF	Bypass mode; reset all clocks and ensure boot mode is active; cb_intosc_hs_div2_clk is used.	cb_intosc_hs_div2_clk (60 to 200 MHz)

I/O Configuration

The flash devices needed for booting must be connected to specific I/Os. The Boot ROM configures these I/Os depending on the flash device selected by boot select setting. To configure the I/Os, the boot ROM performs pin muxing and pin configuration on these I/Os.

The boot ROM is allocated 14 dedicated HPS I/Os. There are three separate I/O tables in the boot ROM, one for each flash device.

On cold reset, the boot ROM always configures the pin muxes and pin configuration. On warm reset, the user has the capability to specify whether or not the boot ROM code configures the I/Os and pin muxes for boot pins after a warm reset. This configuration on warm reset is enabled by programming the romcode_ctrl register in the System Manager.

Refer to the "Boot Source I/O Pins" section for detail on mux select values.

Related Information

- **Boot Source I/O Pins** on page 30-8
- **Hard Processor System I/O Pin Multiplexing** on page 25-1
  Refer to the *Hard Processor Subsystem I/O Multiplexing* chapter for more information regarding booting I/O options and configuration.

L4 Watchdog 0 Timer

The boot ROM enables the L4 Watchdog 0 Timer early in the boot process.

This watchdog is reserved for the boot ROM until the second-stage boot loader indicates that it has started correctly and taken control of the exception vectors. The timeout is at least one second, depending on the

clock select setting. Because the watchdog is reset just before the control passes to second-stage boot loader, the second-stage boot loader must reset the watchdog when it begins execution.

The L4 watchdog 0 timer is reserved for boot ROM use. While booting, if a watchdog reset happens before software control passes to the second-stage boot loader, the boot ROM code attempts to load the last valid second-stage boot loader image, identified by the `initswlastld` register in the System Manager.

If the watchdog reset happens after the second-stage boot loader has started executing but before it writes a valid value to `initswstate` register, the boot ROM increments `initswlastld` and attempts to load that image. If the watchdog reset happens after the second-stage boot loader writes a valid value to `initswstate` register, the boot ROM code attempts to load the image indicated by `initswlastld` register.

Second-Stage Boot Loader

The function of the second-stage boot loader is user-defined. The Altera-provided second-stage boot loader is a combination of initialization, configuration and U-Boot code and contains features such as:

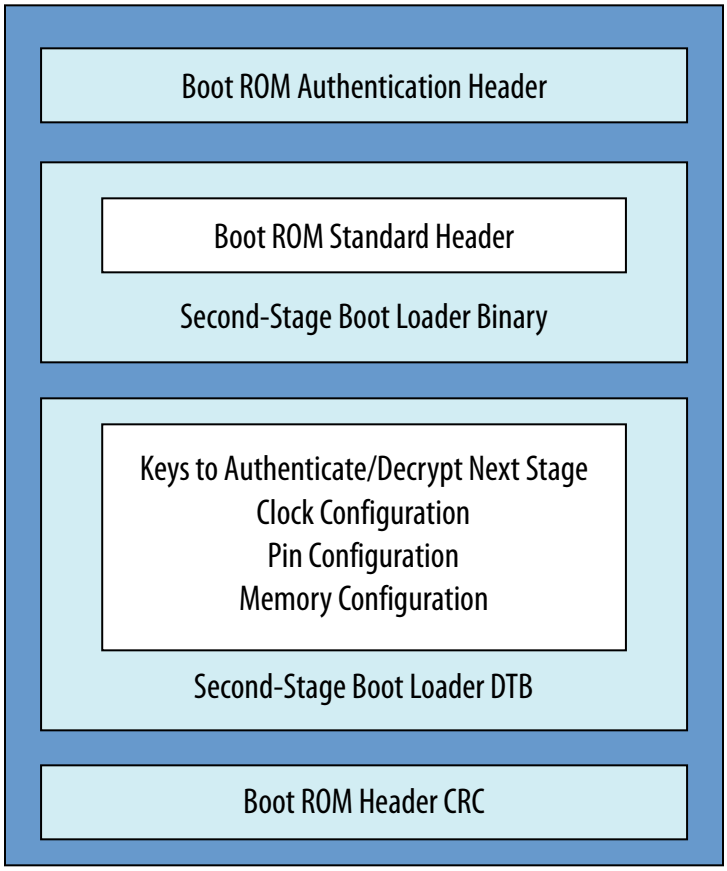
- SD/MMC controller driver
- QSPI controller driver
- Ethernet driver plus protocol support
- Drivers for system-level IP, such as Clock Manager, System Manager, and FPGA Manager
- Cache memory drivers
- UART, timer and watchdog drivers
- FAT file system support
- Flat Image Tree (FIT) image processing
- U-Boot console support including basic essential debug commands
- Cryptographic library
- Flattened device tree (FDT) processing library
- System and memory firewall configuration
- Initialization code for the HPS and FPGA I/O and the interface that loads the next stage of software

If a secure boot is required, the second-stage boot loader can be used to increase the level of security and to authenticate and initiate decryption of the next boot image if necessary.

If the hard memory controller I/O in the SoC device have been configured through the FPGA or HPS and the SDRAM firewall access has been lowered, the second-stage boot loader can be used to load the next stage of the boot software into SDRAM. The maximum length for a second-stage boot loader to fit into on-chip RAM is 208 KB with authentication and 224 KB without authentication. A typical next software stage is loading the application OS software. The second-stage boot loader is allowed to load the next stage boot software from any device available to the HPS. Typical sources include the same flash device that contains the second stage boot loader, a different flash device, or a communication interface such as an EMAC.

If the second-stage boot loader must be authenticated, it must store a public key. Below is a figure which depicts the second-stage boot loader image presented to the boot ROM, during a secure, authenticated boot.

Figure A-8: High-Level Diagram of Second-Stage Boot Loader Image



Secure Boot

The boot ROM supports both non-secure and secure boot.

Secure boot allows the HPS to release from reset into a known state and then validate the authenticity and integrity of the second-stage boot loader code prior to executing it. Secure boot can ensure that only authorized code is executed on the system. In addition, the system has the option to configure the FPGA from the HPS, which provides a secure boot mechanism for the FPGA portion of the SoC. In this mode, the HPS boot code can authenticate the FPGA POF prior to loading it.

The boot ROM determines the security level based on user fuse values that are stored in the Security Manager during initialization. The second-stage boot loader may provide a security header to indicate authentication or decryption for the image. Based on the merged data from security manager and the security header, the boot ROM can load the following images:

- Clear text: No authentication or decryption is required
- Secure, Authenticated - Authentication but no decryption is required on the image
- Secure, Encrypted - Decryption, but no authentication is required. During the decryption process, the boot ROM looks for an encrypted image on the flash device and decrypts it into the on-chip RAM
- Full Secure - Both authentication and decryption are required

The authentication process is independent of the decryption process. However, if authentication and decryption are both required, authentication is done before the decryption process.

For more information regarding Secure Boot, please refer to the *SoC Security* Chapter.

Related Information

SoC Security on page 6-1
For more information about secure boot, refer to the *SoC Security* chapter.

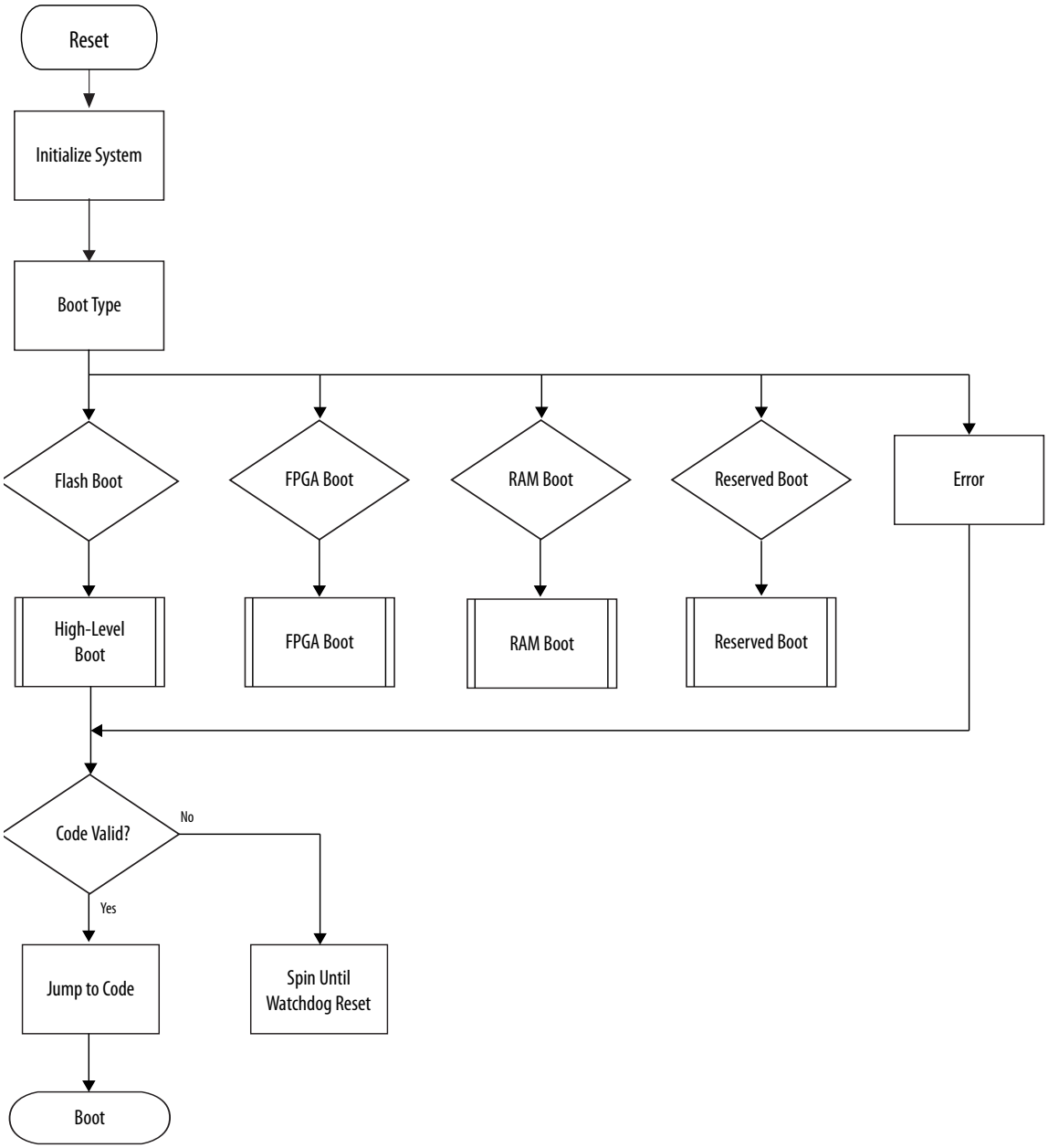
Boot ROM Flow

On a cold reset, the HPS boot process starts when CPU0 is released from reset (for example, on a power up) and executes code in the internal boot ROM at the reset exception address, 0x00000000. The boot ROM code brings the HPS out of reset and into a known state. After boot ROM code is exited, control passes to the next stage of the boot software, referred to as the second-stage boot loader. The second-stage boot loader can be customized and is typically stored external to the HPS in a nonvolatile flash-based memory or in on-chip RAM within the FPGA. The second-stage boot loader can then load an OS, BareMetal application or potentially a third-stage boot loader.

This section describes the software flow from reset until the boot ROM code passes software control to the second-stage boot loader.

The code starts, initializes the system and then depending on the type of boot requested, it may attempt to load the code into the on-chip RAM. An on-chip RAM boot can only occur on a warm reset. Warm boot from on-chip RAM has the highest priority to execute if the `warmram_*` registers in the system manager has been configured to support booting from on-chip RAM on a warm reset. If the `warmram_enable` register in the system manager is set to 0xAE9EFEBC, then the boot ROM code attempts to boot from on-chip RAM on a warm reset and the boot ROM does not configure boot I/Os, pin-muxes or clocks. The `warmram_datastart` and `warmram_length` registers in System Manager allow you to program the offset of the beginning of code and the length of the region of on-chip RAM for CRC validation. If the `warmram_length` register is clear, then the boot ROM does not perform a CRC calculation in the on-chip RAM. If the `warmram_length` register is non-zero, then CRC checking is performed. If CRC checking is successful, then the boot ROM jumps to global address programmed in the `warmram_execution` register. If for three subsequent load attempts, the boot ROM fails to find code or the CRC check fails, the boot ROM spins, waiting for a watchdog reset.

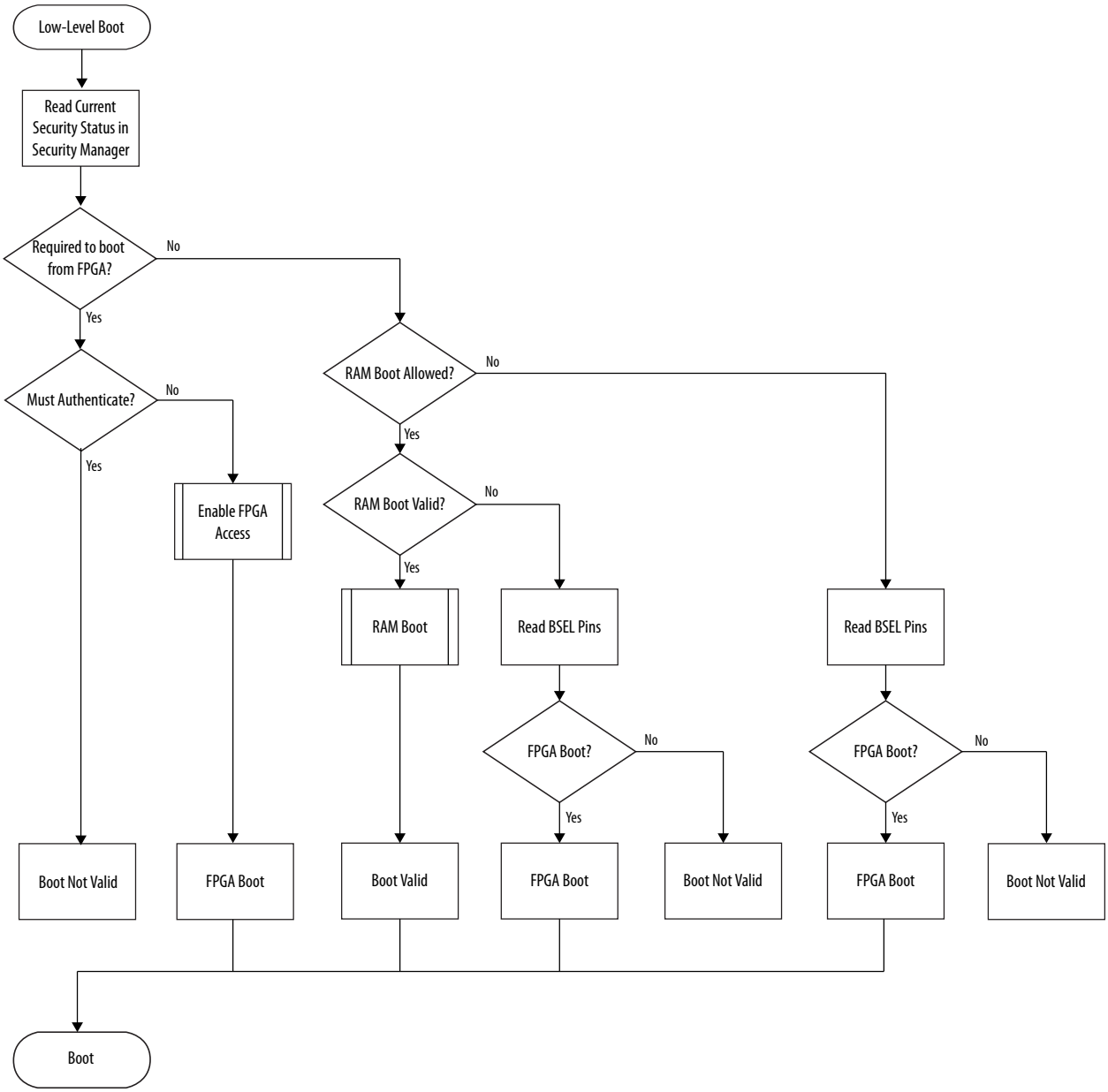
Figure A-9: Main Boot ROM Flow



The boot ROM always executes on CPU0. CPU1 is always held in reset while the main boot ROM code is executing and is only released when required by system software.

As part of determining the boot type, the boot ROM executes a low-level boot flow. The boot ROM code reads the security fuse to determine if the source of the second-stage boot is forced to be the FPGA. If a non-authenticated FPGA boot or a non-CRC on-chip RAM boot is requested or the boot is invalid, it is processed within the low-level boot flow. All other boot types are processed within the high-level boot flow.

Figure A-10: Low-Level Boot Flow

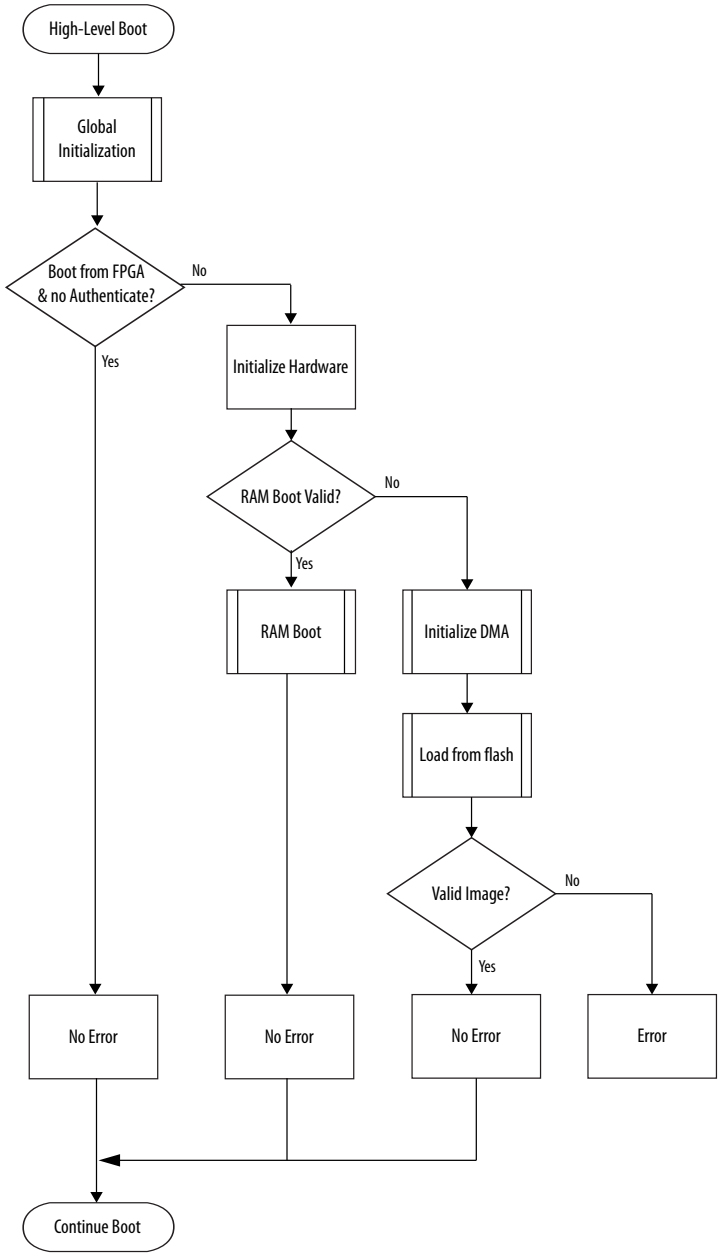


During the low-level portion of the boot ROM flow, the boot ROM reads the security fuses to determine if an FPGA-only boot is required. If so, then the boot ROM must also determine if the fuses indicate that authentication of the POF is needed. If no authentication is required, then a standard FPGA configuration occurs.

If an FPGA-only boot is not required, then the boot ROM checks if an on-chip RAM boot is allowed. If it is, then the boot ROM checks to see if the code is valid. If the code is invalid, the boot ROM reads the BSEL pins to determine if it indicated an FPGA boot.

If the secure fuses indicate that authentication is required for the boot image, then a high-level boot (executed in C code) must be performed.

Figure A-11: High-Level Boot Flow



During the boot process, authentication and decryption can be performed on the boot image. Authentication is independent of decryption; however, if both authentication and decryption are required, then authentication always occurs first. If an authenticated boot is required, then the boot ROM must have a root key to start the authentication process. This key can be implemented in the user fuses, in the FPGA logic elements or as part of the second-stage boot image header. The device configuration fuses determine the source of the key.

During a cold boot from flash memory, the boot ROM code attempts to load the first second-stage boot loader image from flash memory to on-chip RAM and pass control to the second-stage boot loader. If this initial image is invalid, the boot ROM code indexes the `romcode_initswlastld` register and attempts to load the next stored image. The boot ROM will attempt three subsequent loads after the initial one. If there is still no valid image found after the subsequent loads, the boot ROM code checks the FPGA portion of the device for a fallback image.

Note: During the boot process, the boot ROM enables all of the caches (L1 data and instruction caches and L2 cache). If the second-stage boot loader is not loaded from a boot flash device (SD/MMC, QSPI, NAND) properly, the caches may be left on when the boot ROM checks the FPGA portion the device for a fallback image. This situation can lead to issues of coherency when loading code, so caches must be flushed and disabled in the fallback image.

If the warm RAM boot has failed or if a cold reset has occurred, then the boot ROM reads the BSEL value in the `bootinfo` register of the System Manager. If the FPGA is selected as the boot source, then the boot ROM code attempts to execute code at address 0xC0000000 across the HPS-to-FPGA bridge (offset 0x00000000 from bridge). No error conditions are generated if the FPGA does not initialize properly and the watchdog is not enabled for time-out. Instead, the boot ROM continues to wait until the FPGA is available.

If the BSEL bits indicate a boot from external flash, then the boot ROM code attempts to load an image from a flash device into the on-chip RAM, verify and execute it. If the BSEL is invalid or the boot ROM code cannot find a valid image in the flash, then the boot ROM code checks if there is a fallback image in the FPGA. If there is, then the boot ROM executes the fallback image. If there is no fallback image then the boot ROM performs a post-mortem dump of information into the on-chip RAM and awaits a reset.

Note: The acronyms BSEL and BOOTSEL are used interchangeably to define the boot select pins.

The boot ROM code verifies the second-stage boot loader in several ways to ensure a corrupted image is not executed. The first test is of the image header, which identifies the magic number, version, block length, and CRC of the image that protects the block. If any of these are invalid, an error occurs.

Second-Stage Boot Flow

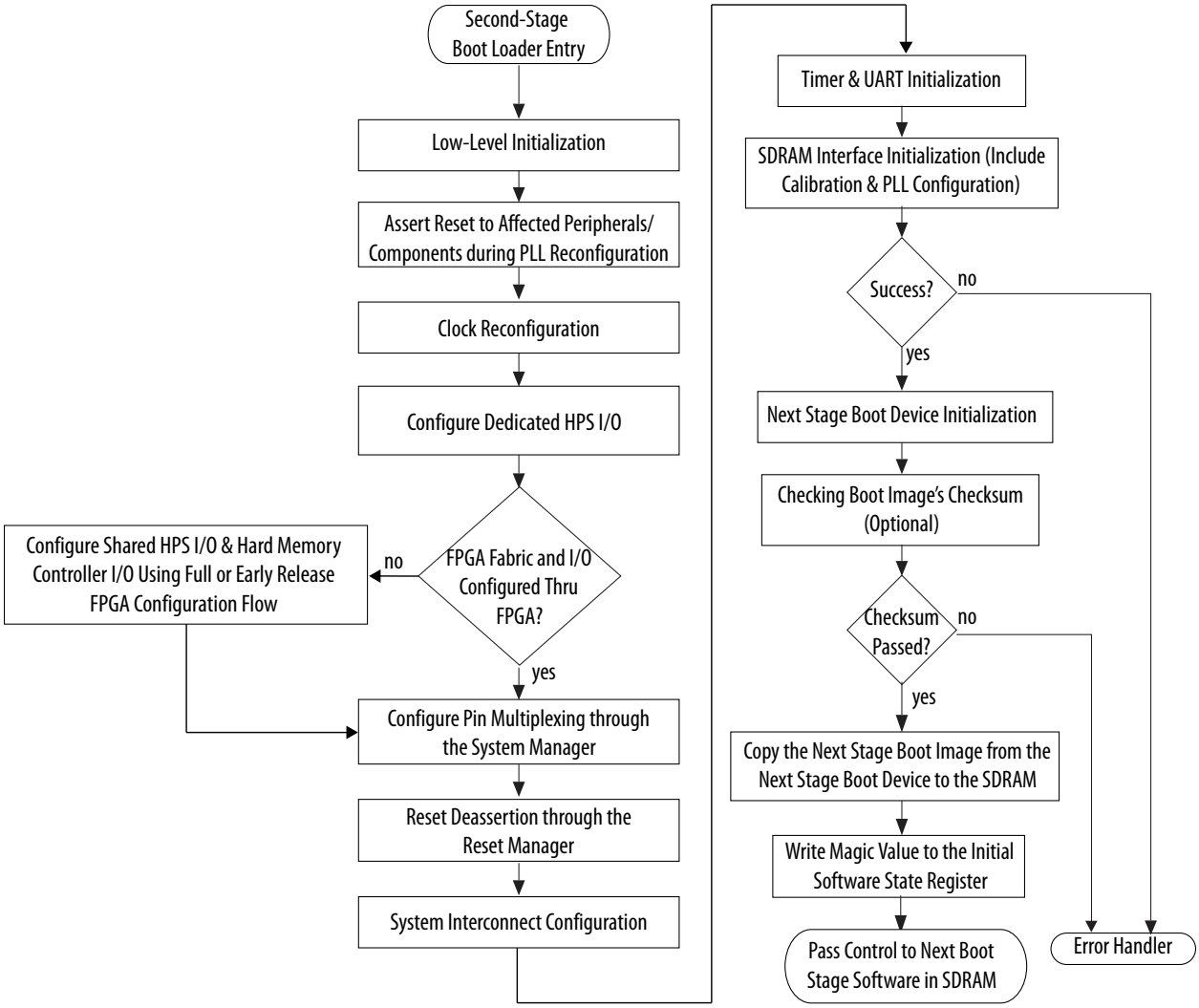
The second stage boot loader has the capability of supporting the following types of boot:

- Non-secure clear text boot
- Secure boot with:
  - Authentication only (also called verified boot)
  - Decryption only
  - Authentication and decryption

Typical Boot Flow (Non-Secure)

A non-secure second-stage boot process typically follows a flow as in the following figure.

Figure A-12: Typical Second-Stage Boot Loader Flow (Non-Secure)



Low-level initialization steps include reconfiguring or disabling the L4 watchdog 0 timer, invalidating the instruction cache and branch predictor, remapping the on-chip RAM to the lowest memory region, and setting up the data area.

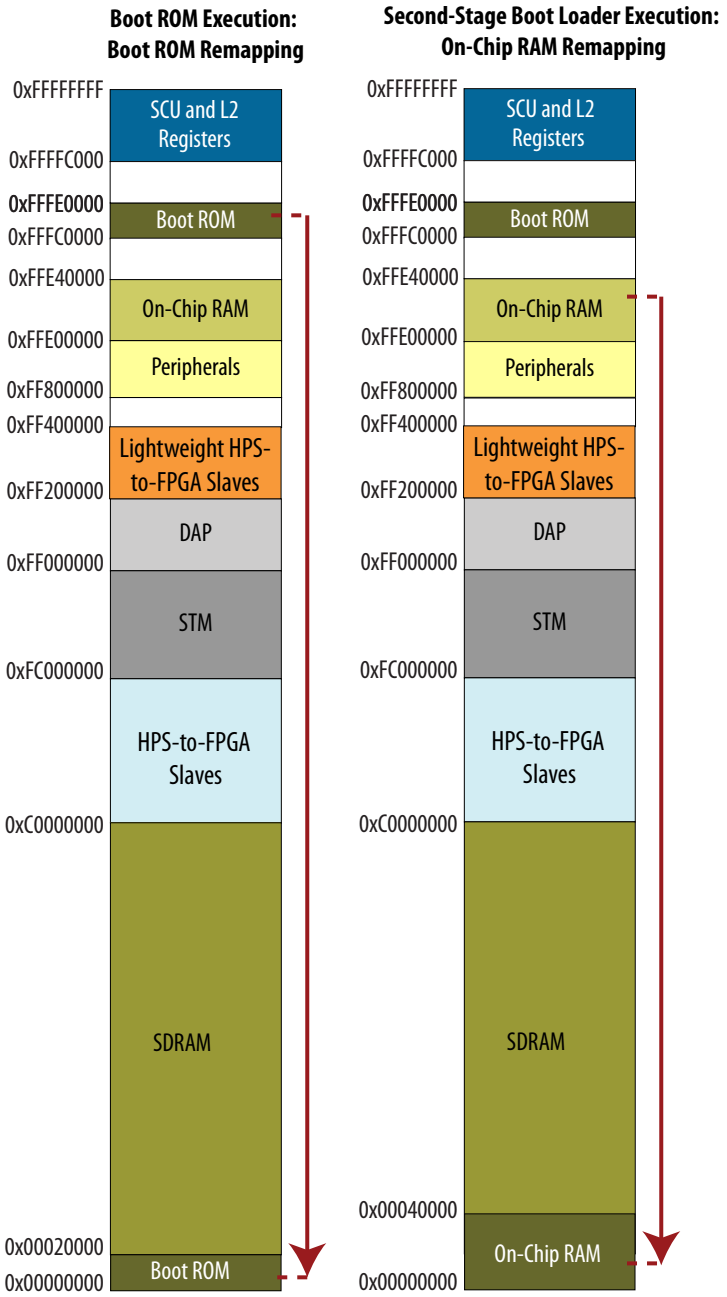
Upon entering the second-stage boot loader, the L4 watchdog 0 timer is active. The second stage boot loader can disable, reconfigure, or leave the watchdog timer unchanged. Once enabled after reset, the watchdog timer cannot be disabled, only paused or reset.

The instruction cache and branch predictor, which were previously enabled by the boot ROM code, must be invalidated. If the second-stage boot loader enables and uses the data cache, it must initialize all levels of the data cache before enabling.

The second-stage boot loader must remap the exception vector table because the exception vectors are still pointing to the exception handler in the boot ROM when it starts executing. By setting the system interconnect remap bit 0 to one, the on-chip RAM mirrors to the lowest region of the memory map. After this remap, the exception vectors use the exception handlers in the boot loader image.

The figure below shows the memory map before and after remap. The first mapping depicts boot ROM execution. The second mapping shows the remap of the on-chip RAM when the remap bit is set during second-stage boot loader execution.

Figure A-13: Remapping the On-Chip RAM



The second-stage boot loader can reconfigure all HPS clocks. During clock reconfiguration, the boot loader asserts reset to the peripherals in the HPS affected by the clock changes.

The I/O assignments for the HPS are configured as part of the IOCSR configuration in the second-stage boot loader. Effectively, a bitstream containing the I/O assignments is sent to the device as part of the

initialization code in the second-stage boot loader. If the FPGA fabric and I/O have not yet been configured through the FPGA and the HPS needs to access SDRAM, you should program the HPS to use the full or early I/O release configuration method to configure the shared and hard memory controller I/O. Refer to "FPGA Configuration" section of the "Booting and Configuration" appendix in the *Arria 10 Hard Processor System Technical Reference Manual* for details on full and early I/O release configuration.

The second-stage boot loader looks for a valid next-stage boot image in the next-stage boot device by checking the boot image validation data and checksum in the mirror image. Once validated, the second-stage boot loader copies the next-stage boot image (OS or application image) from the next-stage boot device to the SDRAM.

Before software passes control to the next-stage boot software, the second-stage boot loader can write a valid value (0x49535756) to the `romcode_initswstate` register in the System Manager. This value indicates that there is a valid boot image in the on-chip RAM. The `romcode_initiswlastld` register holds the index of the last second-stage boot loader software image loaded by the Boot ROM from the boot device. When a warm reset occurs, the Boot ROM loads the image indicated by the `romcode_initswlastld` register if the BSEL value is the same as the last boot.

Related Information

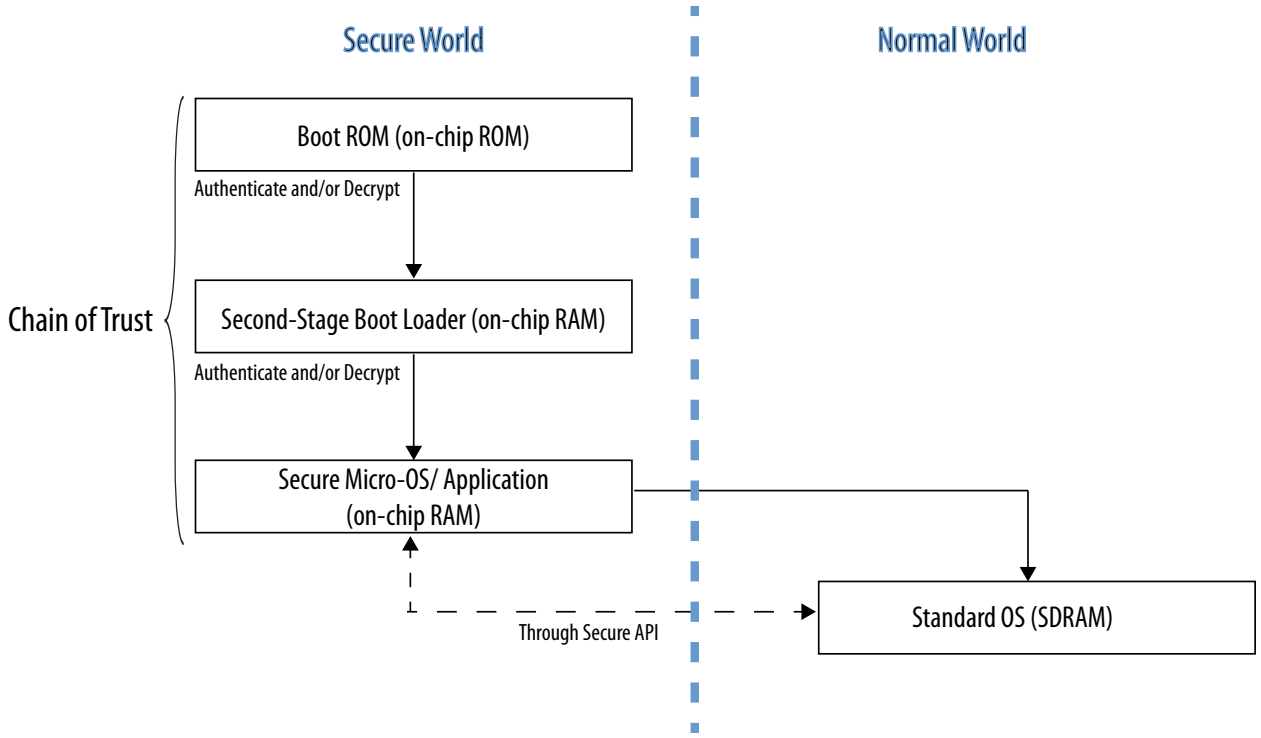
- **FPGA Configuration** on page 30-46
  For more information about full and early I/O release configuration methods
- **Arria 10 SoC Boot User Guide**
  For more information on the steps for booting the Arria 10 SoC

Secure Boot Flow

The main purpose of secure boot is to pass the chain of trust to the subsequent boot software. During a secure boot, the second-stage boot loader may authenticate or decrypt the subsequent boot image, depending on the current state registers in the Security Manager. In addition, the second-stage boot loader

must ensure that the subsequent boot image is executed from secure memory such as on-chip RAM. The second-stage boot loader fits into the chain of trust as such:

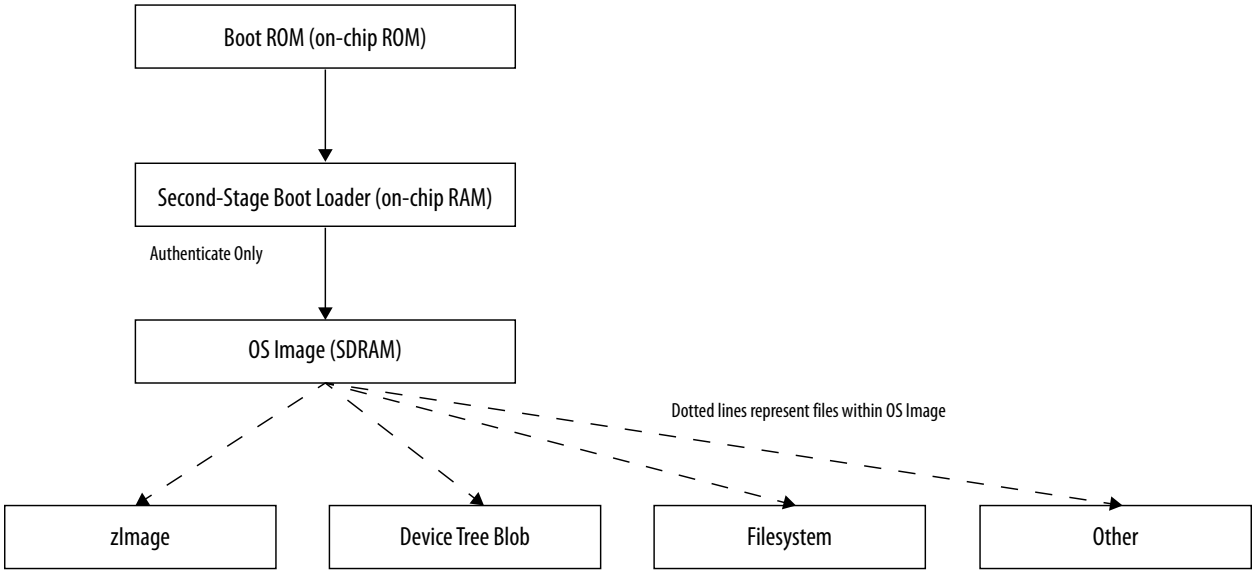
Figure A-14: Secure Boot Flow



The micro OS provides secure APIs to allow the application in the normal world OS to establish trusted services.

During a verified boot, the second-stage boot loader only authenticates the OS image and other images required by the OS. A flow for a verified boot is shown below.

Figure A-15: Verified (Authenticated) Boot Flow



For both the secure and verified boot, the subsequent boot image must be executed in on-chip RAM while the second-stage boot loader is still executing from on-chip RAM. In order to accommodate this requirement, the authentication and decryption process might follow the following steps depicted in the next three diagrams, depending on the type of secure boot chosen.

Figure A-16: Second-Stage Boot Loader Authentication Process

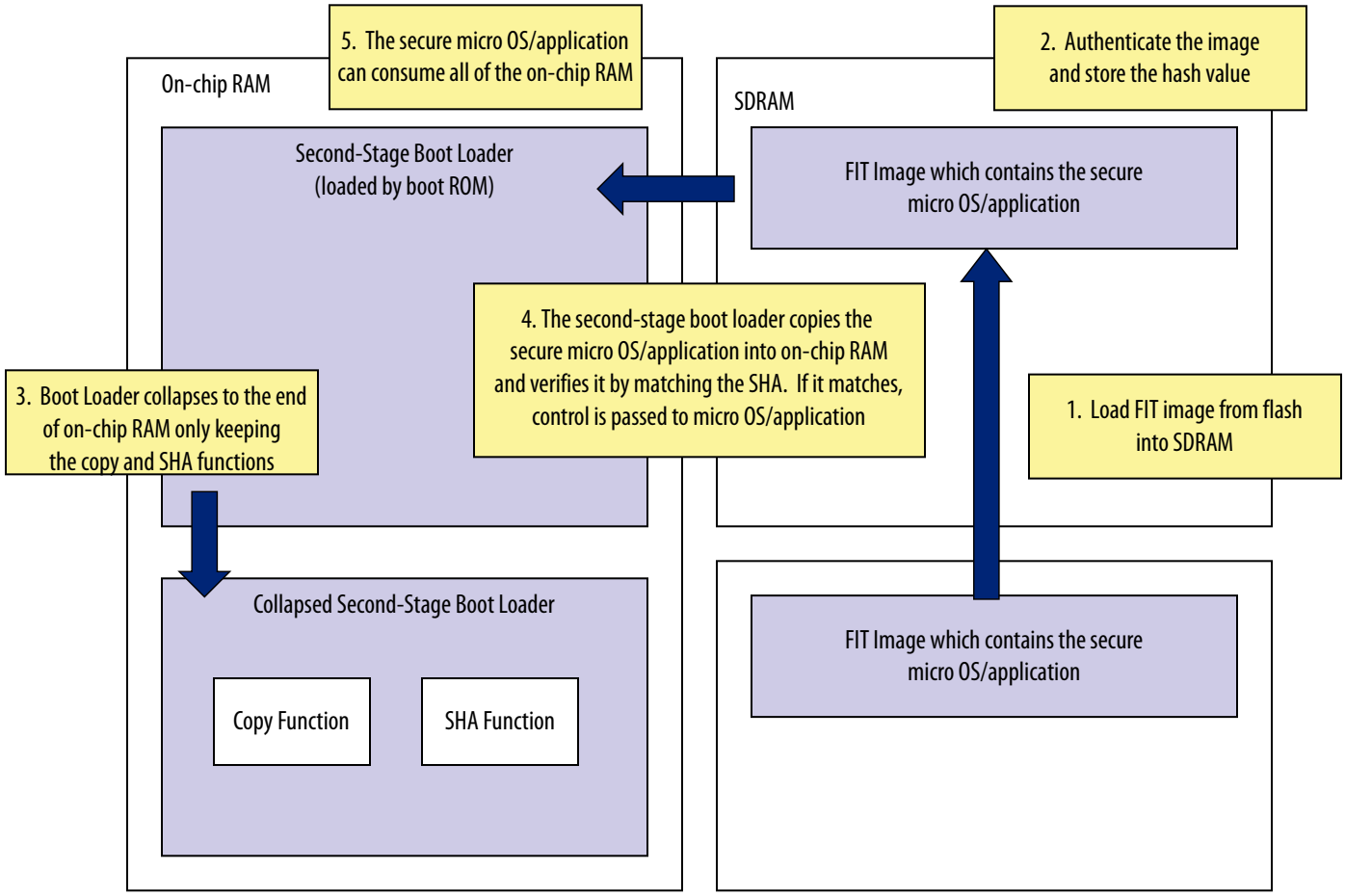
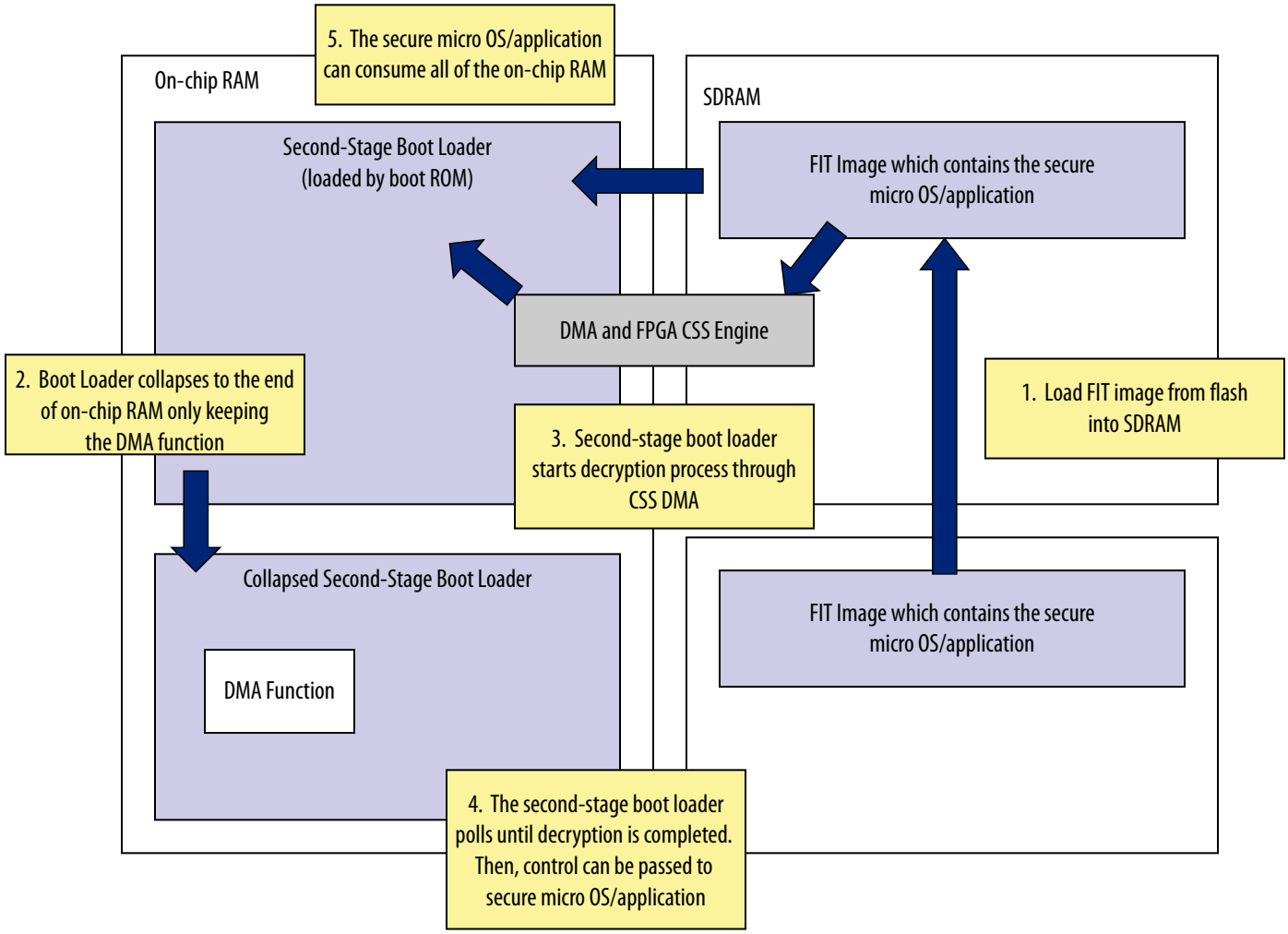
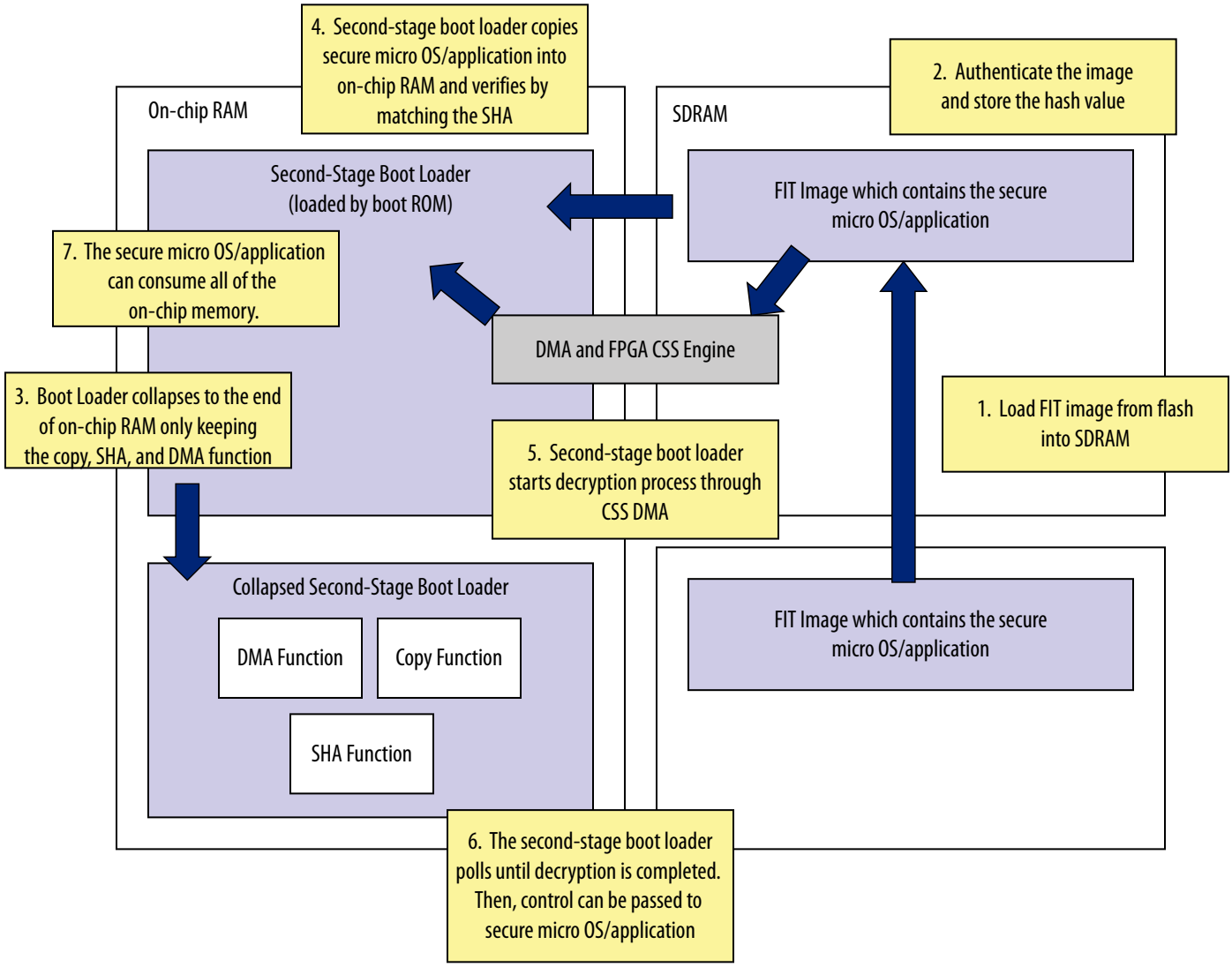


Figure A-17: Second-Stage Boot Loader Decryption Process



Decryption is optional and is not required for secure boot. Upon entry into the second-stage boot loader, the CSS engine is enabled. The second-stage boot loader decrypts the subsequent boot image and disables the CSS engine upon exit.

Figure A-18: Second-Stage Boot Loader Authentication and Decryption Process



HPS State on Entry to the Second-Stage Boot Loader

When the boot ROM code is ready to pass control to the second-stage boot loader, the processor (CPU0) is in the following state:

- Instruction cache is enabled
- Branch predictor is enabled
- Data cache is disabled
- MMU is disabled
- Floating point unit is enabled
- NEON vector unit is enabled
- Processor is in ARM secure supervisor mode

The boot ROM code sets the ARM Cortex-A9 MPCore registers to the following values:

- r0—contains the pointer to the shared memory block, which is used to pass information from the boot ROM code to the second-stage boot loader. The shared memory block is located in the top 4 KB of on-chip RAM.
- r1—contains the length of the shared memory.
- r2—unused and set to 0x0.
- r3—points to the version block.

All other MPCore registers are undefined.

Note: When booting CPU0 using the FPGA boot, or when booting CPU1 using any boot source, all MPCore registers, caches, the MMU, the floating point unit, and the NEON vector unit are undefined. HPS subsystems and the PLLs are undefined.

When the boot ROM code passes control to the second-stage boot loader, the following conditions also exist:

- CPU0 is in a secure system mode.
- The boot ROM is still mapped to address 0x0.
- The L4 watchdog 0 timer is active and has been toggled.
- The shared L2 cache is left in an uninitialized state.

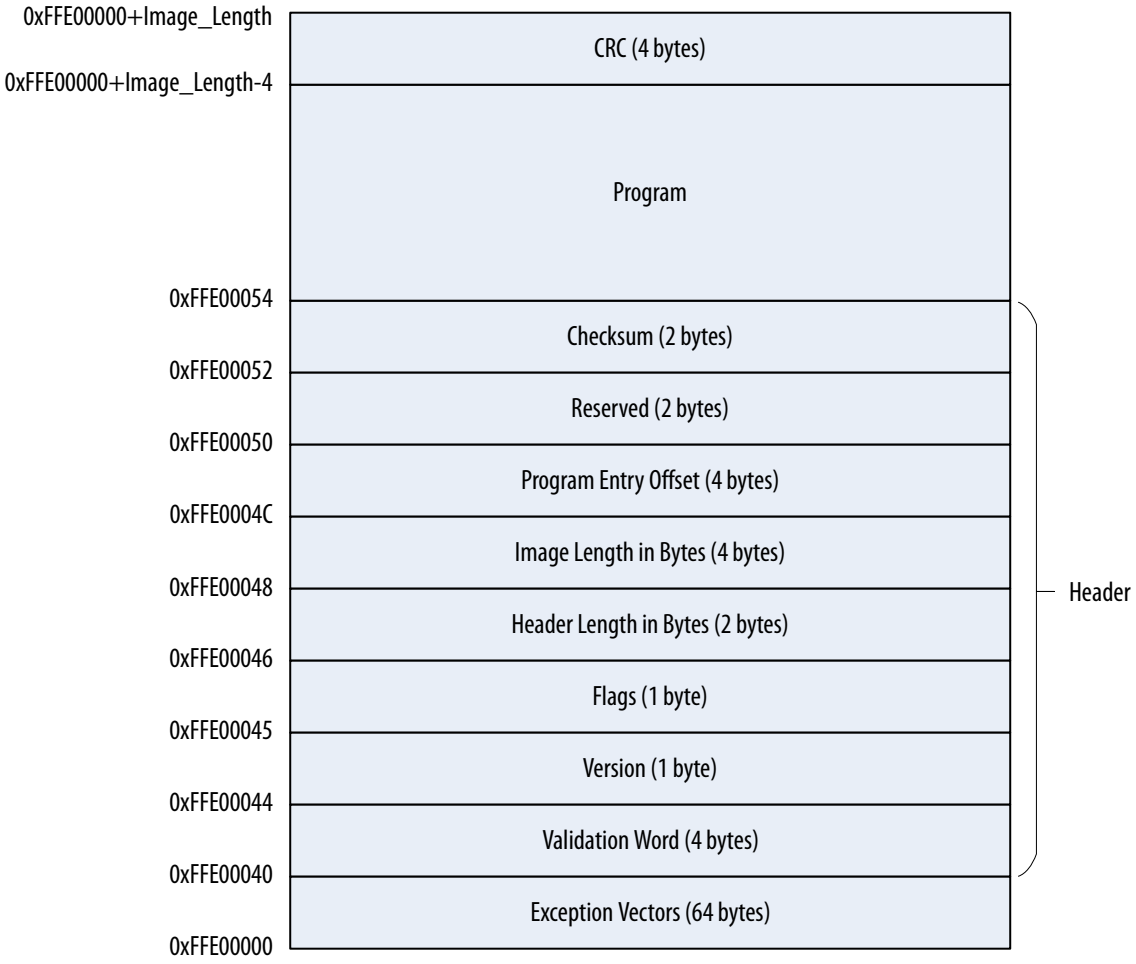
The boot ROM also saves the state of the reset cause in the `stat` register of the Reset Manager, and this information is available for the Second-Stage Boot Loader to read.

Loading the Second-Stage Boot Loader Image

The boot ROM code loads the image from flash memory into the on-chip RAM and passes control to the second-stage boot loader. The boot ROM code checks for a valid image by verifying the header and cyclic redundancy check (CRC) in the second-stage boot loader image.

The maximum second-stage boot loader size is 208 KB with authentication and 224 KB without. This size is limited by the on-chip RAM size of 256 KB, where 48KB or 32 KB is reserved as a workspace for the boot ROM data and stack depending on whether authentication is enabled or not. The second-stage boot loader can use this reserved region (for its stack and data, for example) after the boot ROM code passes control to the second-stage boot loader. This reserved region is overwritten by the boot ROM code on a subsequent reset. The following figure shows the second-stage boot loader image layout in the on-chip RAM after being loaded from the boot ROM.

Figure A-19: Second-Stage Boot Loader Image



- **Exception vectors**—Exception vectors are located at the start of the on-chip RAM. Typically, the second-stage boot loader remaps the lowest region of the memory map to the on-chip RAM (from the boot ROM) to create easier access to the exception vectors.
- **Header**—contains information such as validation word, version, flags, program length, security settings,and checksum for the boot ROM code to validate the second-stage boot loader image before passing control to the second-stage boot loader.
- **Version**—This byte describes the version and layout of the program header. This value must be 0x1.
- **Entry point**—After the boot ROM code validates the header, the boot ROM code jumps to this address.
- **Flags**— Not used.
- **Header length**— The length of the header in bytes.
- **Image length** — The total length of the image (including the exception vectors and the CRC field)
- **Program entry offset**— The boot ROM jumps to this offset. The offset value is in bytes from the start of the program header (0xFFE00040).
- **Reserved** —Not used. Must be set to 0x0.

- **Checksum**— A simple checksum of all the bytes from 0xFFE00040 to 0xFFE00051, inclusive
- **Program** — Program contents which typically includes U-Boot binary and U-Boot Device Tree Blob (DTB).
- **CRC**—There is a CRC value for the entire image. The CRC consists of data located from address 0xFFFE0000 to 0xFFFE0000+(Image Length)–0x0004. The polynomial used to validate the image is:

  $$x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^{8} + x^{7} + x^{5} + x^{4} + x^{2} + x + 1$$

  There is no reflection of the bits. The initial value of the remainder is 0xFFFFFFFF and the final value is XORed with 0xFFFFFFFF.

FPGA Configuration

You can configure the FPGA portion of the SoC device with non-HPS sources or by utilizing the HPS.

When the HPS handles the initial FPGA configuration, software executing on the HPS writes the configuration image to the FPGA Manager in the HPS. Software can control the configuration process and monitor the FPGA status by accessing the control and status register (CSR) interface in the FPGA Manager.

For the HPS to configure the FPGA, the mode select (MSEL) pins must be set to passive mode (fast or slow). The following table shows the pin encodings for MSEL[2:0] that are available to the HPS. Refer to the *Configuration, Design Security, and Remote System Upgrades in Arria 10 Devices* document for a comprehensive list of MSEL encodings.

Table A-16: MSEL Pin Settings Options for FPGA Configuration through HPS

Configuration Scheme allowed through HPS	Power-On Reset (POR) Delay	Valid MSEL[2:0]
FPP (x16, and x32)	Fast	000
	Standard	001

Note: You must also select the configuration scheme in the **Configuration** page of the **Device and Pin Options** dialog box in the Quartus Prime software. Based on your selection, the option bit in the programming file is set accordingly.

There are two types of initial FPGA configuration flows you can choose from when using the HPS to configure the FPGA.

- Full FPGA configuration flow: In this flow, the FPGA fabric, the shared I/O and hard memory controller I/O are configured. After full configuration is complete, the hard memory controller I/O, shared I/O and FPGA I/O are available to the HPS.
- Early I/O release configuration flow: In this flow, all I/O are configured and the HPS shared I/O and hard memory controller I/O are released so that the HPS has immediate access to them. At a later time, the FPGA fabric is configured and the FPGA I/O are released and available to the user design.

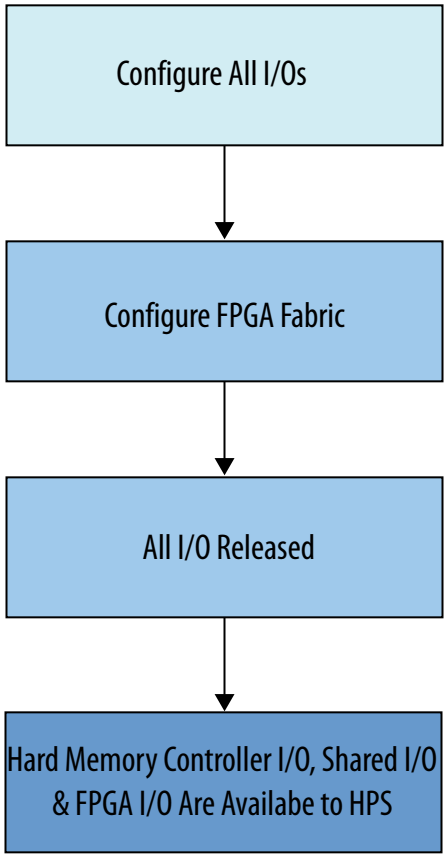
Related Information

- **FPGA Manager** on page 4-1
  For more information regarding programming the FPGA through the HPS, refer to the *FPGA Manager* chapter.
- **Configuration, Design Security, and Remote System Upgrades in Arria 10 Devices**

Full FPGA Configuration Flow Through HPS

Figure A-20: Full FPGA Configuration Flow Through HPS

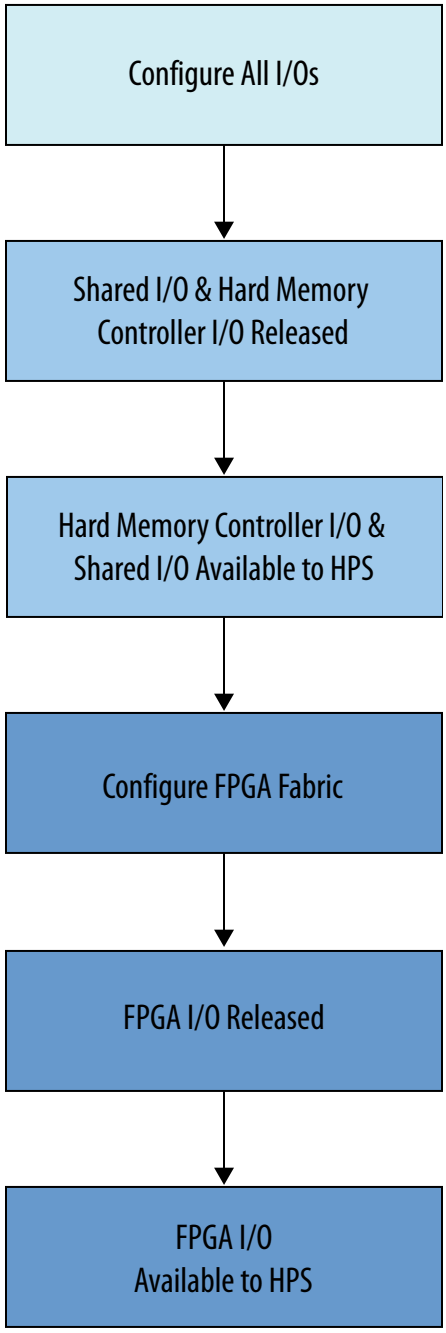
The FPGA fabric and the all I/Os are configured first and then all the I/O are released for use by the HPS.



Early I/O Release FPGA Configuration Flow Through HPS

Figure A-21: Early I/O Release FPGA Configuration Flow Through HPS

The I/Os are configured and the shared and hard memory controller I/Os are released so that the HPS has immediate access to them. The FPGA fabric configuration follows at a later time.

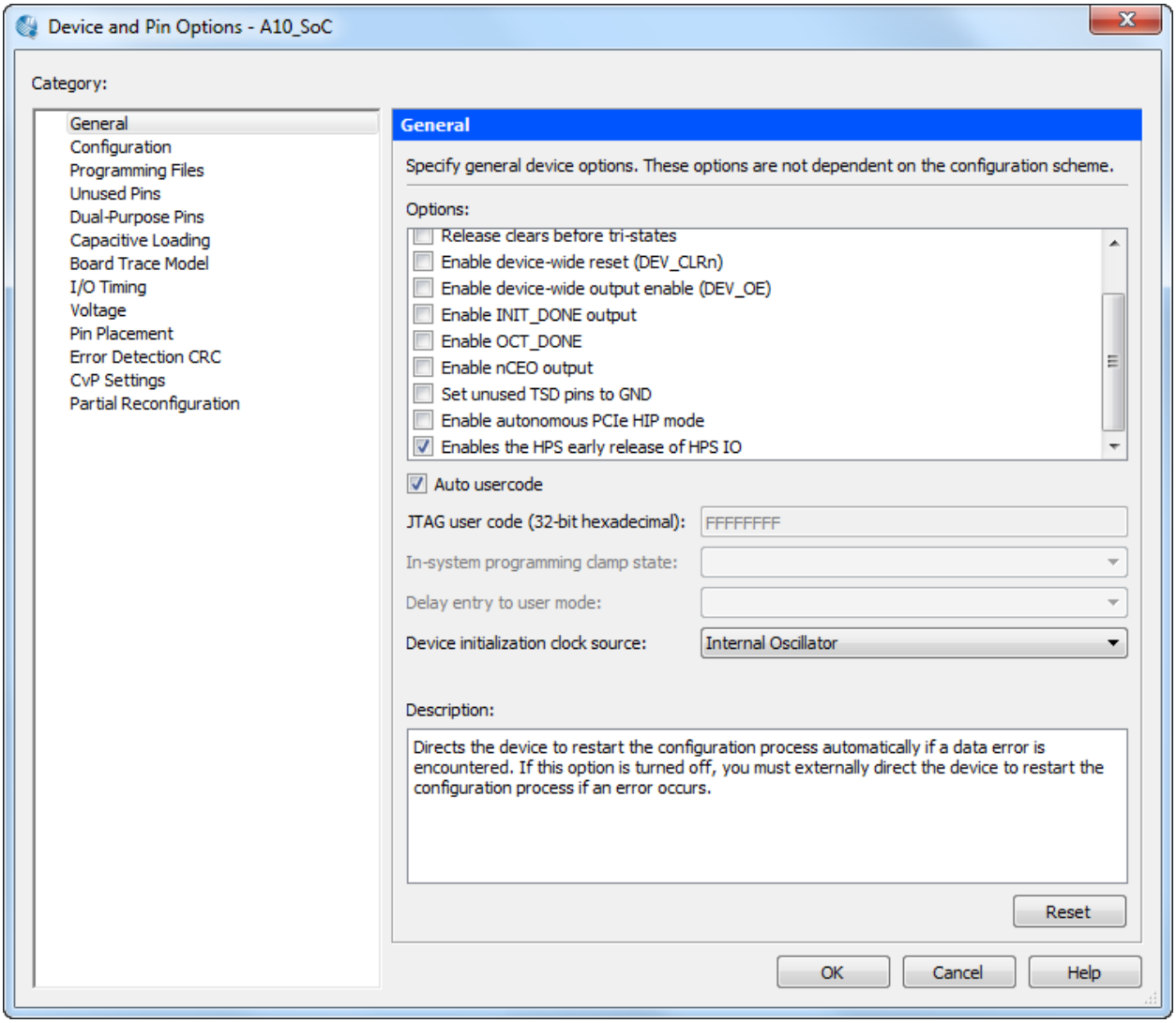


Note: The FPGA fabric is not required to be programmed immediately after the first I/O configuration. You can also program the I/Os, re-program the I/Os and then program the FPGA fabric as long as the early I/O release configuration option is enabled in each I/O configuration file.

You can enable early I/O configuration in Quartus Prime:

1. Click on the the **Assignments** menu and select **Device**.
2. When the **Device settings** window opens, click the **Device and Pin Options** button.
3. In the **General** category, click on the option labeled "Enables the HPS early release of HPS IO".

Figure A-22: Quartus Prime Early I/O Release Setting



Handling an FPGA Configuration Failure after Early I/O Release

If an FPGA configuration failure occurs after early I/O release, the HPS shared I/O and hard memory controller I/O enter input tri-state mode. In this situation, any accesses to peripherals or memory interfacing these I/O will fail. HPS dedicated I/O are not affected by FPGA configuration failures.

If you are using early I/O release and issuing an HPS-initiated FPGA configuration afterward, you must ensure that your software routine includes a way to avoid FPGA configuration failure or a recovery mechanism that does not rely on the HPS shared I/O or hard memory controller I/O.

You must implement one of the following solutions in software.

- To prevent FPGA configuration failure, load the FPGA core configuration file into SDRAM and perform an FPGA integrity check before initiating configuration. For example, you can run a custom integrity check, such as a CRC, on the `.rbf` file to validate that there are no errors in the file. Altera also recommends that you enable ECC capabilities to avoid bit-stream corruption issues.
- To recover from an FPGA configuration failure, execute the HPS-initiated configuration from on-chip RAM. If the HPS is executing entirely from on-chip RAM, you can recover your application by reconfiguring the peripheral `.rbf` file to reenable the HPS shared I/O and hard memory controller I/O. After these I/O are recovered, your application can either attempt to configure the same FPGA image that caused the original failure or use a fallback core `.rbf` file to prevent the same failure from occurring repeatedly.

Note: If you are using full FPGA configuration, you do not need to make any of the adjustments listed above. For full FPGA configuration, the .rbf file contains both the core and peripheral configuration and your application can only access the HPS I/O after the device is successfully configured.

Arria 10 SoC FPGA Configuration Sequence Through FPGA Manager

The FPGA Manager in the HPS has the capability to execute an initial FPGA configuration using the full or early I/O release flow.

The steps below detail the programming of the FPGA Manager to initiate configuration.

1. Read the `f2s_msel[2:0]` field in the `imgcfg_stat` register to verify that the configuration mode is the passive fast (000) or passive slow (001).
2. Write the `cfgwidth` bit of the `imgcfg_ctrl_02` register in the FPGA manager to match the characteristics of the configuration image. For full, standard configuration, the `cfgwidth` bit must be set to 1. You can only initiate a full, standard configuration in 32-bit passive parallel mode.
3. Configure the `cdratio` field. The `cdratio` is a function of the datawidth and whether the POF in encrypted or compressed. This information should be provided through the Quartus Prime configuration. For example,if `cfgwidth` is 32-bits and the POF is compressed, you must program the `cdratio` field to 0x8. If the POF is encrypted, then program the `cdratio` to 0x4.
4. Ensure there are no other external devices interfering with the FPGA programming. Poll the `f2s_nconfig_pin` and `f2s_nstatus_pin` bits from the `imgcfg_stat` register until they are both 1.
5. Program the following bits to deassert signal drives from the HPS before overriding the signals.
   - `s2f_nce` =1 in the `imgcfg_ctrl_01` register
   - `s2f_pr_request`= 0 in the `imgcfg_ctrl_01` register
   - `en_cfg_ctrl`= 0 in the `imgcfg_ctrl_02` register
   - `s2_nconfig` = 1 in the `imgcfg_ctrl_00` register
   - `s2f_nstatus_oe` = 0 in the `imgcfg_ctrl_00` register
   - `s2f_condone_oe` = 0 `imgcfg_ctrl_00` register
6. Enable overrides for the `DATA`, `DCLK`, `nCE`, `PR_REQUEST` and `nCONFIG` signals.
   - `s2f_nenable_config` = 0 in the `imgcfg_ctrl_01` register
   - `s2f_nenable_nconfig` = 0 in the `imgcfg_ctrl_00` register
7. Disable overrides for the `nSTATUS` and `CONF_DONE` signals.

- s2f_nenable_nstatus = 1 in the imgcfg_ctrl_00 register
- s2f_nenable_condone = 1 in the imgcfg_ctrl_00 register

8. Assert the chip select signal.
   - s2f_nce=0 in the imgcfg_ctrl_01 register
9. Repeat step 4. Ensure there are no other external devices interfering with the FPGA programming. Poll the f2s_nconfig_pin and f2s_nstatus_pin bits from the imgcfg_stat register until they are both 1.
10. Reset the configuration.
    a. Write a 0 to the s2f_nconfig bit in the imgcfg_ctrl_00 register.
    b. Poll the f2s_nstatus_pin bit in the imgcfg_stat register until it is clear.
    c. Write a 1 to the s2f_nconfig bit in the imgcfg_ctrl_00 register.
    d. Poll the f2s_nstatus_pin bit in the imgcfg_stat register until it is set. In addition, read and confirm the f2s_condone_pin is clear and the f2s_condone_oe is set in the imgcfg_stat register.
11. Enable DCLK and the DATA path by setting the en_cfg_ctrl bit in the imgcfg_ctrl_02 register.
12. Write the bitstream data to the img_data_w register. When sending the bitstream data, you can periodically read and confirm that the f2s_nstatus_pin bit is set in the imgcfg_stat register. If it is not set, configuration has failed and you must restart with step 1.
13. When you have completed writing the data, continuously poll the f2s_condone_pin bit in the imgcfg_stat register until it reads as 1. When this bit is 1, it indicates configuration has completed. If the routine times out with f2s_nstatus_pin=0, configuration has failed and you must restart from step 1.
14. Write the dclkcnt register with 0xF and continuously poll until the dclkcntstat register reads as 0x0. A minimum of 2 DCLK cycles are needed after the CONFIG_DONE assertion for initialization to start.
15. Wait for the initialization sequence to complete. Poll the f2s_usermode bit in the imgcfg_stat register until it reads as 1.
16. Disable the DATA path and DCLK by clearing the en_cfg_ctrl bit in the imgcfg_ctrl_02 register.
17. Disable the chip select. Set the s2f_nce bit in the imgcfg_ctrl_01 register.
18. Disable overrides to the nCONFIG, DATA and DCLK signals.
    - s2f_nenable_config=1
    - s2f_nenable_nconfig=1
19. Check that user mode is enabled and the configuration is done by checking that the bits below have the following values:
    - f2s_usermode=1 in the imgcfg_stat register
    - f2s_nstatus_pin=1 in the imgcfg_stat register
    - f2s_condone_pin=1 in the imgcfg_stat register

FPGA Reconfiguration

The FPGA Manager also supports FPGA reconfiguration through two methods: full or partial reconfiguration.

Full FPGA Reconfiguration

A full FPGA reconfiguration requires a system reboot. The following table details the supported reconfiguration options:

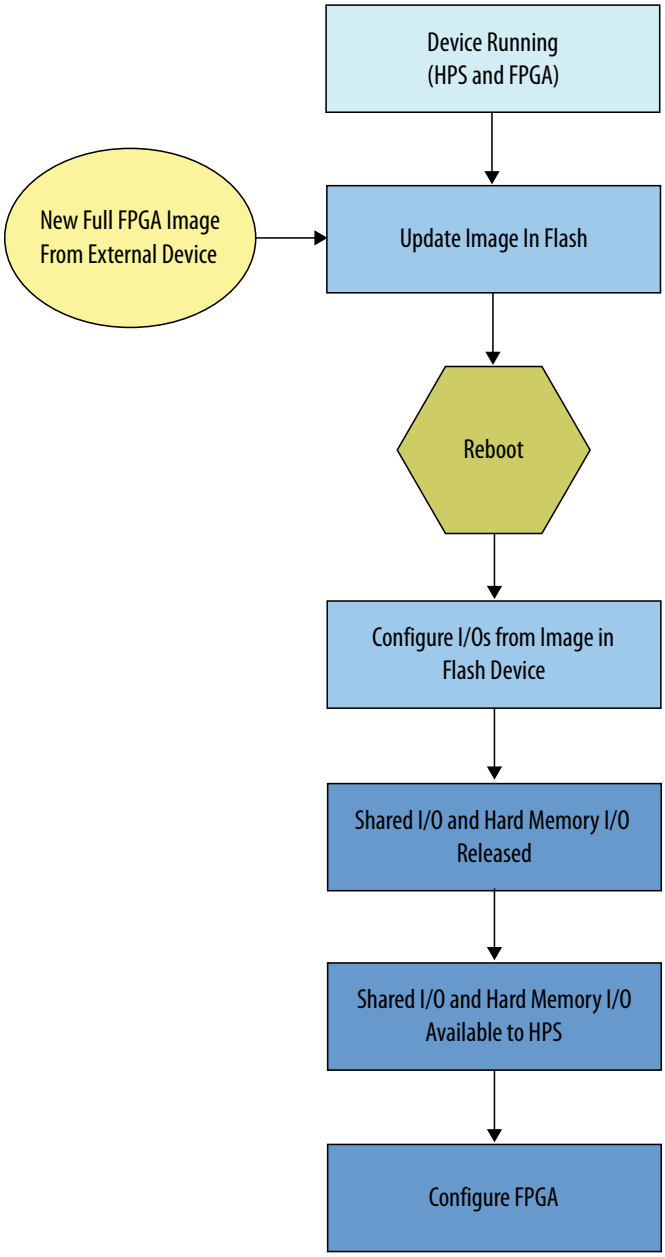
Table A-17: Supported Reconfiguration Options

Reconfiguration Option	Configuration Method	HPS Reboot Required
FPGA Core Only	Partial Reconfiguration	No
FPGA and/or HPS Shared I/O[64]	Full Reconfiguration	Yes
FPGA Core + FPGA I/O + HPS Shared I/O[64]	Full Reconfiguration	Yes

If a new FPGA fabric image is required and performing a system reboot is acceptable, then you can update the FPGA image in flash and perform a full system reboot of both hardware and software before following the steps found in the "Arria 10 SoC FPGA Configuration Sequence Through FPGA Manager."

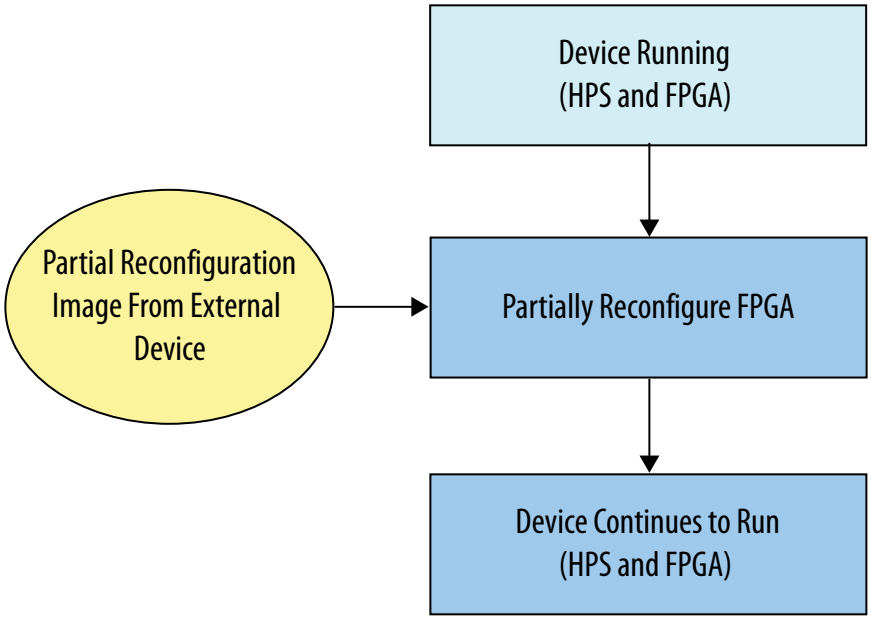
[64] HPS dedicated I/O are not affected by full reconfiguration.

Figure A-23: Full FPGA Fabric Reconfiguration with System Reboot



If a new FPGA fabric image is required and performing a system reboot is unacceptable because the hard memory controller or shared I/O are being used by the HPS, then you can perform a reconfiguration using the available partial reconfiguration support. For the partial reconfiguration sequence, follow the steps found in the "Arria 10 SoC FPGA Partial Reconfiguration Sequence Through FPGA Manager" section.

Figure A-24: Full FPGA Fabric Reconfiguration without System Reboot



Related Information

- **Arria 10 SoC FPGA Configuration Sequence Through FPGA Manager** on page 30-50
- **Arria 10 SoC FPGA Partial Reconfiguration Sequence Through FPGA Manager** on page 30-54

Arria 10 SoC FPGA Partial Reconfiguration Sequence Through FPGA Manager

Partial reconfiguration allows you to reconfigure part of the device while other sections remain running. This implementation divides the design in to a static region and a partial reconfiguration region.

- Static Region: Comprised of hard memory controller I/O, shared I/O, FPGA I/O and a portion of your FPGA design
- Dynamic Region: Remaining portion of FPGA design

The HPS performs partial reconfiguration while the FPGA portion of the device is in user mode. The following sequence suggests one way for software to perform a partial configuration:

- If the `f2s_pr_ready` bit in the `imgcfg_stat` register of the FPGA Manager is set, continue to step 7.
- If `f2s_pr_ready` bit in the `imgcfg_stat` register of the FPGA Manager is clear, then go back and repeat step 5. Note that a minimum of 16 `DCLK` pulses are required.

If an HPS warm reset occurs in the middle of a partial reconfiguration, software must repeat the steps for partial configuration. After an HPS cold reset, software must repeat the steps for **Arria 10 SoC FPGA Configuration Sequence Through FPGA Manager** on page 30-50.

1. Read the `f2s_usermode` bit of the `imgcfg_stat` register in FPGA Manager to ensure that the FPGA is in user mode.
2. Read the `f2s_msel[2:0]` field in the `imgcfg_stat` register to verify that the configuration mode is the passive fast (000) or passive slow (001). The FPGA manager only supports passive parallel programming.
3. Set the `cdratio` bit of the `imgcfg_ctrl_02` register to match the characteristics of the partial reconfiguration image and configure the `cfgwidth` bit of the `imgcfg_ctrl_02` register to 0. Partial reconfiguration only supports 16-bit passive parallel programming.
4. Prepare for partial reconfiguration. Set the `en_cfg_ctrl` bit of the `imgcfg_ctrl_02` register to give the FPGA manager control of the configuration input signals.
   a. Read and confirm that `s2f_pr_request=0` in the `imgcfg_ctrl_01` register and `s2f_nconfig=1` in the `imgcfg_ctrl_00` register.
   b. Drive the chip select signal by clearing the s2f_nce bit in the `imgcfg_ctrl_01` register.
   c. Enable overrides for configuration data and control by setting the `en_cfg_ctrl` bit in the `imgcfg_ctrl_02` to 1.
   d. Disable overrides that are not used for partial reconfiguration by programming the following bits in the `imgcfg_ctrl_00` register:
      - `s2f_condone_oe` = 0
      - `s2f_nstatus_oe` = 0
      - `s2_nconfig` = 1
      - `s2f_nenable_condone` = 1
      - `s2f_nenable_nstatus` = 1
      - `s2f_nenable_nconfig` = 0
5. Enable HPS to override the `DATA`, `DCLK`, `nCE` and `PR_REQUEST` signals to the CSS by clearing the `s2f_nenable_config` bit in the `imgcfg_ctrl_01` register.
6. Run `DCLK` to clear any errors. Write a value of 0x100 to the `dclkcnt` register to generate 256 `DCLK` cycles. Poll the `dclkstat` register until `dcntdone=1`. You must implement a timeout in your software loop that either exits with an error or restarts at step 1 if `dcntdone` is not set after a period of time.
7. Initiate a partial reconfiguration request.
   a. Set the `s2f_pr_request` bit of the `imgcfg_ctrl_01` register to assert `PR_REQUEST`
   b. Write 0x7FF to the `dclkcnt` register.
   c. Poll the `dclkstat` register until `dcntdone=1` .Again, you must implement a timeout in your software loop that either exits with an error or restarts at step 1 if `dcntdone` is not set after a period of time.
8. Poll the `f2s_pr_ready` bit and the `f2s_pr_error` bit in the `imgcfg_stat` register. When `f2s_pr_ready=1`, the `PR_READY` signal is asserted. Alternately, the FPGA Manager can be configured to generate an interrupt when the `PR_READY` signal is asserted. If `f2s_pr_error=1`, you cannot continue with partial reconfiguration because security is preventing the software from updating the configuration. Altera recommends that your software routine implement an exit with a timeout error when `f2s_pr_ready` and `f2s_pr_error` continue to read as 0 after ten register reads.

9. When the PR_READY signal is asserted, write the partial reconfiguration image to the img_data_w register in the FPGA Manager. You can also choose to use a DMA to transfer the configuration image from a peripheral device to the FPGA manager.
10. Poll the imgcfg_stat register to observe the f2s_pr_done and f2s_pr_error bits. When the f2s_pr_done bit is set, partial FPGA reconfiguration is complete. If f2s_pr_done or f2s_pr_error is not set to 1 after reading the imgcfg_stat register ten times, generate a timeout error and continue on to the next step.
11. Clear the s2f_pr_request bit of the imgcfg_ctrl_01 register to deassert PR_REQUEST.
12. Poll the dcntdone bit of the dclkstat register until it reads as 1, which indicates that all the DCLKs have been sent.
13. Write a 1 to the dcntdone bit of the dclkstat register to clear the completed status flag.
14. Disable the DATA path and DCLK by clearing the en_cfg_ctrl bit of the imgcfg_ctrl_02 register.
15. Disable the chip select. Set the s2f_nce bit in the imgcfg_ctrl_01 register.
16. Disable overrides to the nCONFIG, DATA and DCLK signals by setting the s2f_nenable_config in the imgcfg_ctrl_01 register.
17. Check that user mode is enabled and the configuration is done by checking that the bits below have the following values:
    - f2s_usermode=1 in the imgcfg_stat register
    - f2s_nstatus_pin=1 in the imgcfg_stat register
    - f2s_condone_pin=1 in the imgcfg_stat register

Related Information

Arria 10 SoC FPGA Configuration Sequence Through FPGA Manager on page 30-50

Document Revision History

Table A-18: Document Revision History

Date	Version	Changes
October 2016	2016.10.28	Modified the "Remapping the On-Chip RAM" diagram in the "Typical Boot Flow (Non-secure)" section
May 2016	2016.05.27	• Added *Handling an FPGA Configuration Failure after Early I/O Release* section. • Updated "Second-stage Boot Loader Image Layout" figure in the *Loading the Second-Stage Boot Loader Image* section. • Changed the name of the internal QSPI reference clock from qspi_clk to qspi_ref_clk; and the external QSPI output clock, from sclk_out to qspi_clk.

Date	Version	Changes
May 2016	2016.05.03	• Updated SD/MMC device clock values in the *CSEL Settings for SD/MMC Controller* section. • Updated configuration sequence steps in the *Arria 10 SoC FPGA Configuration Sequence Through FPGA Manager* section. • Updated the reconfiguration sequence steps in the *Arria 10 SoC FPGA Partial Reconfiguration Sequence Through FPGA Manager* section. • Added bus mode to the "SD/MMC Controller Default Settings" table in the *Default Settings of the SD/MMC Controller* section.
December 2015	2015.12.11	Updated the Full FPGA Reconfiguration section with the supported reconfiguration options.
November 2015	2015.11.02	Added Quartus Prime setting information to "Early I/O Release FPGA Configuration Through HPS" section
June 2015	2015.06.12	• Updated "Typical Second-Stage Loader Flow (Non-Secure)" figure in "Typical Boot Flow (Non-Secure)" section • Added content to "FPGA Overview" • Described I/O types and added details to booting option figures in "Booting and Configuration Options" • Added table to "Boot Fuses" section • Updated the CSEL setting tables in the "CSEL Settings for NAND Controller" section • Updated the "Quad SPI Controller Default Settings" table in the "Quad SPI Controller Default Settings" section • Updated the CSEL setting tables in the "Quad SPI Controller CSEL settings" section • Updated "Low-Level Boot Flow" figure in the "Typical Boot Flow (Non-Secure)" section • Added content regarding configuration flows in "FPGA Configuration" section • Added "Full FPGA Configuration Flow Through HPS" section • Added "Early I/O Release FPGA Configuration Flow Through HPS" section • Added "FPGA Reconfiguration" section with "Full FPGA Reconfiguration Section"

Date	Version	Changes
May 2015	2015.05.04	• Added more detail in the "FPGA Configures First" table of the "Booting and Configuration Options" section. • Added alternate "Boot Source Mux Selects" table in "Boot Source I/O Pins" section and "I/O Configuration" section. • Added more detail regarding booting from FPGA in the "Boot Select" section. • Added note in the "Boot ROM Flow" section about caches being enabled by the Boot ROM.

December 2014	2014.12.15	• Removed "Boot Stages" section.
		• The following sections were added:
		  • "Boot ROM Flow"
		  • "Typical Second-Stage Boot Flow"
		  • "HPS State on Entry to the Preloader"
		  • "Loading the Second-Stage Boot Image"
		  • "FPGA Configuration" section with "Arria 10 SoC FPGA Full Configuration" and "Arria 10 SoC FPGA Partial Reconfiguration"
		• In the "Boot Definitions" section, "Boot Source I/O Pins", "Boot Fuses", and "Flash Memory Device" subsections were added.

August 2014	2014.08.18	Initial release